Journal of Computational Physics 496 (2024) 112520

Contents lists available at ScienceDirect

Journal of Computational Physics

journal homepage: www.elsevier.com/locate/jcp

Data-driven Whitney forms for structure-preserving control volume analysis

Jonas A. Actor [a], Xiaozhe Hu [b], Andy Huang [c], Scott A. Roberts [d], Nathaniel Trask [a,*]

[a] *Center for Computing Research, Sandia National Laboratories, 1515 Eubank SE, Albuquerque, NM 87123, USA*
[b] *Department of Mathematics, Tufts University, 177 College Ave., Medford, MA 02155, USA*
[c] *Radiation and Electrical Science, Sandia National Laboratories, 1515 Eubank SE, Albuquerque, NM 87123, USA*
[d] *Engineering Sciences Center, Sandia National Laboratories, 1515 Eubank SE, Albuquerque, NM 87123, USA*

ARTICLE INFO

Keywords:
Scientific machine learning
Whitney forms
Finite element exterior calculus
Partition of unity
Data-driven modeling

ABSTRACT

Control volume analysis models physics via the exchange of generalized fluxes between subdomains. We introduce a scientific machine learning framework adopting a partition of unity architecture to identify physically-relevant control volumes, with generalized fluxes between subdomains encoded via Whitney forms. The approach provides a differentiable parameterization of geometry which may be trained in an end-to-end fashion to extract reduced models from full field data while exactly preserving physics. The architecture admits a data-driven finite element exterior calculus allowing discovery of mixed finite element spaces with closed form quadrature rules. An equivalence between Whitney forms and graph networks reveals that the geometric problem of control volume learning is equivalent to an unsupervised graph discovery problem. The framework is developed for manifolds in arbitrary dimension, with examples provided for $H(\mathrm{div})$ problems in $\mathbb{R}^2$ establishing convergence and structure preservation properties. Finally, we consider a lithium-ion battery problem where we discover a reduced finite element space encoding transport pathways from high-fidelity microstructure resolved simulations. The approach reduces the 5.89M finite element simulation to 136 elements while reproducing pressure to under 0.1% error and preserving conservation.

1. Overview and background

In recent years a number of works aim to use machine learning (ML) to develop data-driven models and surrogates for physical systems. Broadly, these use data either: to develop predictive models where first-principles derivation is prohibitively expensive or complex; or to learn a surrogate model for expensive/complex simulations to facilitate tasks requiring repeated evaluation of a forward model (e.g., uncertainty quantification, optimization, inverse modeling). In multiscale modeling, many seek data-driven homogenization techniques to represent mesoscale geometry at continuum scales while avoiding oversimplifications of geometry [49,54,80,59,14,37,24,27]. While ML has provided a number of remarkable new capabilities in these fields, the rigorous numerical analysis typical of traditional finite element (FEM) simulation is generally lacking. ML-based data-driven models are typically unable to provide the guaranteed convergence, stability, and preservation of mathematical/physical structure that form the cornerstone of verification and validation. In more classical contexts, even traditional reduced order models (ROMs) struggle with

* Corresponding author.
E-mail address: natrask@sandia.gov (N. Trask).

https://doi.org/10.1016/j.jcp.2023.112520
Received 16 October 2022; Received in revised form 20 September 2023; Accepted 22 September 2023

Available online 16 October 2023
0021-9991/Published by Elsevier Inc.

structure-preservation [62], requiring problem specific remedies to treat conservation, geometric structure, inf-sup conditions, bounds preservation, and other mathematical/physical structure [43,22,3,56,34,45]. A further challenge is how to maintain performance for nonlinear problems; using projection as a basis of model reduction requires incorporation of techniques such as hyper-reduction [17] to mitigate computational scaling with respect to full-order problem.

The work presented here employs concepts from structure-preserving finite element discretization and discrete exterior calculus [7,15] to obtain data-driven models and surrogates providing such guarantees by construction. *Physics-informed ML* and related techniques [42,58,38] incorporate domain knowledge by augmenting loss functions with physics-based regularizers. While successful for many problems, these techniques generally enforce physics by penalty so that properties are achieved only within optimization error. As a result, mathematical analysis and physical requirements often require complex training strategies with ad hoc hyperparameter calibration to achieve predictive results [77]. In contrast, we propose a *structure-preserving ML* approach in which the neural architecture itself is engineered to preserve properties by construction, allowing rigorous analysis.

Classically, mimetic or structure-preserving discretizations of partial differential equations (PDEs) are broadly characterized via strong form or variational frameworks. The discrete exterior calculus (DEC) forms the basis for the former, allowing analysis of finite volume methods ubiquitous in continuum mechanics and transport problems [20,52,53]. Finite element exterior calculus (FEEC) forms the basis for the latter [8,6], providing variational generalizations. Both frameworks admit a unified analysis due to their shared algebraic/topological structure [15]: while DEC defines discrete operators acting directly on differential forms associated with a mesh, FEEC defines finite element spaces which interpolate differential forms. Both achieve structure preservation by preserving discrete analogs of the generalized Stokes theorem to construct a *de Rham complex* - a commuting diagram encoding preservation of topological structure related to vector div/grad/curl, providing an abstract framework for analyzing: stability; conservation; and non-trivial null-spaces related to e.g., involution in electromagnetism or incompressibility in mechanics. Through mass lumping one may recover DEC schemes from FEEC [61]; we will exploit this connection in the data-driven setting.

In our recent work [71], we developed a data-driven discrete exterior calculus (DDEC) which allows training of a parameterized de Rham complex associated with machine learnable div/graph/curl defined on graph data. This recasts message-passing graph neural networks (GNNs) in the language of numerical analysis: the usual message passing and aggregation steps [74] may be interpreted as learning generalized fluxes associated with conservation balances. This alternative perspective provides Hodge decompositions, Poincare inequalities, and other tools supporting numerical analysis, and learns nonlinearities directly on the reduced model circumventing the need for hyper-reduction. This framework has a drawback however, in that one must define a priori a graph which represents well the underlying domain. For engineering of network systems (e.g., electrical circuits, hydraulic networks, discrete fracture networks, thermal circuits, mechanical spring/damper models) this is natural, but for general full field data associated with continuum mechanics and transport this is restrictive. This drawback is shared by other recent works pursuing GNNs as a means of learning physics [65,30,55]. The current work instead performs *graph discovery* in an unsupervised manner by identifying physically relevant control volumes which support a traditional integral control volume analysis, without reference to a traditional mesh.

In traditional FEEC, the low-order Whitney forms are constructed from the barycentric interpolant of differential forms associated with nodes, edges, faces and cells of a mesh. Barycentric interpolants form a partition of unity (POU) - namely, they form a set of positive functions which sum to unity, partitioning space into non-disjoint subdomains. Broadly, POUs are a tool to localize analysis and approximation: in differential geometry they are the fundamental tool for constructing an atlas of charts [67,32], while in numerical analysis they have been used to perform refinements and construct localized approximation spaces [10,21,78,25,44,51]. In previous work we have developed deep learning architectures to discover POUs which admit optimal piecewise polynomial approximation of data, providing a meshfree ML approximation which exhibits hp-convergence for regression tasks [46,70]. Here, we show that by replacing barycentric coordinates with a machine learnable partition of unity in the traditional Whitney form construction, we may construct discrete coboundary operators associated with these partitions. This allows an unsupervised discovery of control volumes encoding conservation balances. The resulting FEM space may then be used in concert with an equality constrained optimization procedure to discover structure-preserving reduced-order models (ROMs) from data.

These ROMs are particularly attractive in their natural treatment of nonlinearities. In traditional projection-based ROMs, one obtains terms which depend upon a *known* full-order model when testing against nonlinearities, mandating techniques such as hyperreduction to maintain computational tractability. In contrast, in DDEC we postulate a parameterized (potentially nonlinear) ROM *without knowledge* of a full-order model, instead using data to select parameters. In [71] we provide a complete theory and computational examples for the general nonlinear case. While structure-preservation may still be obtained while applying hyperreduction to projection-based ROMs [22,33,47], the current approach may offer attractive alternatives, particularly for applications with “hidden physics” (e.g., closures, homogenization, and aleatoric uncertainty) where the governing equations are unknown. For ease of exposition when introducing the Whitney form theory, we restrict focus to linear problems and postpone consideration of nonlinear multiphysics problems to a future work.

The paper is organized as follows. We first gather mathematical preliminaries and definitions in Section 2. For brevity, we defer to our previous work for a comprehensive introduction to the exterior calculus [71]. We review barycentric coordinates, partitions of unity, and the construction of Whitney forms. In Section 3 we present a construction of data-driven Whitney forms in the case of Euclidean geometry, before generalizing to an arbitrary manifold in Section 4. The former requires knowledge only of basic vector calculus and linear algebra. In the latter, repeating the derivation with discrete exterior calculus highlights the intimate connections between the graph, discrete, and finite element exterior calculus. In Section 5 we demonstrate how this may be used as a framework for learning physics, developing a novel architecture for parameterizing the POU. While several aspects of the construction are necessary to guarantee well-posedness of the resulting model, most notable is the fact that the finite element space admits closed form expressions for quadrature, allowing backpropagation through the physics model to the partitions without introducing a variational crime. We then consider the application of the framework to $div - grad$ problems, demonstrating convergence for smooth problems and compatible treatment of problems with discontinuous coefficients. Finally we consider an application building a surrogate for electrostatics in a Lithium-ion battery. A high-fidelity microstructure conforming FEM simulation consisting of 5.89 million elements is used as training data, which we reproduce to under 1% error using 8 data-driven Whitney forms. This provides a structure-preserving reduced-order model with learnt Whitney forms identifying transport pathways through microstructure. In contrast to traditional multiscale methods working with representative volume elements or other simplifications of microstructure geometry, the benchmark provides a flux-conservative characterization of as-built microstructure which may be obtained from computed tomography scans.

2. Mathematical background

2.1. Exterior algebra and differential forms

We first briefly recall some definitions and basic results from exterior algebra and differential forms, introducing the abstract framework before specializing to the finite element exterior calculus (FEEC) setting. A *graded vector space* is a vector space V that can be expressed as $V = \bigoplus_{k=-\infty}^{\infty} V_k$, where V_k are subspaces. Introducing the linear map $\partial_k : V_k \to V_{k-1}$ called the *boundary operator*, with the property $\partial_k \circ \partial_{k+1} = 0$, we define a *chain complex* as follows

$$\begin{tikzcd} \cdots \arrow[r] & V_{k+1} \arrow[r, "\partial_{k+1}"] & V_k \arrow[r, "\partial_k"] & V_{k-1} \arrow[r] & \cdots, \end{tikzcd}$$

where we refer to elements $\Omega \in V_k$ as *chains* and associate with each a real-valued *cochain* $\omega \in V^k$. We adopt the convention of using subscripts to denote chains (V_k) and superscripts for cochains (V^k). Given *coboundary* maps $\mathsf{d}^k : V^k \to V^{k+1}$ satisfying $\mathsf{d}^k \circ \mathsf{d}^{k-1} = 0$, and *codifferential maps* $d^k : V^{k+1} \to V^k$ satisfying $d^{k-1} \circ d^k = 0$, we finally arrive at the following primal and dual *cochain complexes*.

$$\begin{tikzcd} \cdots \arrow[r] & V^{k-1} \arrow[r, "\mathsf{d}^{k-1}"] & V^k \arrow[r, "\mathsf{d}^k"] & V^{k+1} \arrow[r] & \cdots \end{tikzcd} \tag{1}$$

$$\begin{tikzcd} \cdots & V^{k-1} \arrow[l] & V^k \arrow[l, "d^{k-1}"'] & V^{k+1} \arrow[l, "d^k"'] & \cdots \arrow[l] \end{tikzcd} \tag{2}$$

In the specializations of the abstract theory, the coboundary/codifferential operators will define discrete derivatives between mesh/graph entities. This abstract perspective will allow us to derive a formula relating the graph exterior calculus, discrete exterior calculus and finite element exterior calculus at the end of Section 4; we exploit this connection to set up a learning problem on physics data. For a complete introduction we refer readers to [71] for an overview of DEC, to [8] for FEEC, to [15] for a unification of DEC/FEEC, and to [9,1] for an overview of mathematical background.

Let $\Omega \subset \mathbb{R}^d$ be a domain and $\Lambda(\Omega)$ be the exterior algebra with exterior product $\wedge$. We specifically consider the exterior algebra of differential forms and use $\Lambda^k(\Omega)$ to denote the space of smooth differential k-forms on Ω. Introducing the exterior derivative $\mathsf{d}^k : \Lambda^k(\Omega) \to \Lambda^{k+1}(\Omega)$ satisfying $\mathsf{d}^k \circ \mathsf{d}^{k-1} = 0$, we have the well-known de Rham complex

$$\begin{tikzcd} 0 \arrow[r] & \Lambda^0(\Omega) \arrow[r, "\mathsf{d}^0"] & \Lambda^1(\Omega) \arrow[r, "\mathsf{d}^1"] & \cdots \arrow[r, "\mathsf{d}^{d-1}"] & \Lambda^d(\Omega) \arrow[r] & 0 \end{tikzcd}$$

One important property of the exterior derivative is the so-called Leibniz rule

$$\mathsf{d}(w \wedge v) = (\mathsf{d}w) \wedge v + (-1)^k w \wedge (\mathsf{d}v), \quad \forall\, w \in \Lambda^k(\Omega),\ v \in \Lambda^j(\Omega) \tag{3}$$

For the special case of $\Omega \subset \mathbb{R}^3$ with Lipschitz boundary $\partial\Omega$, the de Rham complex may be expressed in standard vector calculus terminology

$$\begin{tikzcd} 0 \arrow[r] & C^\infty(\Omega) \arrow[r, "\mathrm{grad}"] & {[C^\infty(\Omega)]^3} \arrow[r, "\mathrm{curl}"] & {[C^\infty(\Omega)]^3} \arrow[r, "\mathrm{div}"] & C^\infty(\Omega) \arrow[r] & 0. \end{tikzcd}$$

It is a complex due to the exact sequence property that $\mathrm{curl} \circ \mathrm{grad} = \mathrm{div} \circ \mathrm{curl} = 0$. In the finite-element exterior calculus we introduce Sobolev spaces $H(\mathsf{D}, \Omega) := \{v \in L^2(\Omega),\ \mathsf{D}v \in L^2(\Omega)\}$, $\mathsf{D} = \mathrm{grad}, \mathrm{curl}$, and div. To treat boundary conditions we consider Sobolev spaces with homogeneous Dirichlet boundary conditions, i.e., $H_0(\mathsf{D}, \Omega) := \{v \in H(\mathsf{D}, \Omega),\ \mathrm{tr}\, v = 0 \text{ on } \partial\Omega\}$, $\mathsf{D} = \mathrm{grad}, \mathrm{curl}$, and div, with corresponding de Rham complex

$$\begin{tikzcd} 0 \arrow[r] & H_0(\mathrm{grad}, \Omega) \arrow[r, "\mathrm{grad}"] & H_0(\mathrm{curl}, \Omega) \arrow[r, "\mathrm{curl}"] & H_0(\mathrm{div}, \Omega) \arrow[r, "\mathrm{div}"] & L^2(\Omega) \arrow[r] & 0. \end{tikzcd} \tag{4}$$

This provides the FEEC specialization of the primal cochain complex in Equation (1). A fundamental theorem of exterior calculus is the generalized Stokes theorem,

$$\int_\Omega \mathsf{d}w = \int_{\partial\Omega} \mathrm{tr}\, w, \qquad w \in \Lambda^{d-1}(\Omega). \tag{5}$$

Replacing w with $u \wedge v$ in (5) and applying the Leibniz rule (3), we obtain the integration-by-parts formula

$$\int_\Omega (\mathsf{d}u) \wedge v = (-1)^{k+1} \int_\Omega u \wedge (\mathsf{d}v) + \int_{\partial\Omega} \mathrm{tr}(u \wedge v), \quad v \in \Lambda^k(\Omega),\ u \in \Lambda^{d-k-1}(\Omega). \tag{6}$$

By combining the primal cochain complex (Equation (4)) with the integration-by-parts formula, we arrive at the following dual de Rham complex

$$0 \longleftarrow L^2(\Omega) \xleftarrow[-\operatorname{div}]{} H(\operatorname{div},\Omega) \xleftarrow[\operatorname{curl}]{} H(\operatorname{curl},\Omega) \xleftarrow[-\operatorname{grad}]{} H(\operatorname{grad},\Omega) \longleftarrow 0. \tag{7}$$

Note that the dual de Rham complex has no boundary conditions.

2.2. Partition of unity and barycentric coordinates

Consider a domain $\Omega \subset \mathbb{R}^d$ with an open cover U_i. A *partition of unity* (POU) $\Phi = \{\phi_i(\boldsymbol{x})\}$ is a collection of finitely many smooth functions $\phi_i : \Omega \to [0,1]$ such that $\phi_i(\boldsymbol{x}) \geq 0$, $\operatorname{supp}(\phi_i) \subseteq U_i$, and

$$\sum_i \phi_i(\boldsymbol{x}) = 1, \quad \forall \boldsymbol{x} \in \Omega. \tag{8}$$

For the purposes of this work, we assume Ω to be compact and, for computational tractability, that the cardinality of Φ is finite.

Applying the exterior derivative d to this expression, we have

$$\sum_i \mathsf{d}\phi_i(\boldsymbol{x}) = 0, \quad \forall \boldsymbol{x} \in \Omega. \tag{9}$$

When $\Omega = K$ is a *simplex* whose vertices are $\boldsymbol{v}_0, \boldsymbol{v}_1, \cdots, \boldsymbol{v}_d$, a particular example of a partition of unity is the set of *barycentric coordinates* $\{\lambda_i\}_{i=0}^d$, which are the unique linear polynomials satisfying (8) and (9) as well as the Kronecker delta property $\lambda_i(\boldsymbol{v}_j) = \delta_{ij}$. In addition, barycentric coordinates reproduce linear polynomial spaces, i.e., for any linear polynomial $f : K \to \mathbb{R}$, we have

$$f(\boldsymbol{x}) = \sum_i f(\boldsymbol{v}_i)\lambda_i(\boldsymbol{x}), \tag{10}$$

Consequentially, we also have

$$\sum_i \boldsymbol{v}_i \lambda_i(\boldsymbol{x}) = \boldsymbol{x} \tag{11}$$

$$\sum_i \boldsymbol{v}_i \otimes \mathsf{d}\lambda_i(\boldsymbol{x}) = \boldsymbol{I}. \tag{12}$$

When Ω is a convex d-dimensional polytope with vertex set $\boldsymbol{v}_i$, one can define *generalized barycentric coordinates* λ_i as a set of non-negative functions satisfying (10) for any linear polynomial data. In this case, λ_i still form a partition of unity and the properties (9), (11), and (12) still hold. In this case however, the choice of λ_i is not unique and they may no longer consist of linear polynomials. There are many approaches to define them, e.g., Wachspress coordinates [75], mean value coordinates [26], and moving least squares coordinates [50].

In this paper, we will consider a class of parameterized POUs which may be learned from data. In our recent work [70] we have shown that data-driven POUs may be used to localize polynomial approximation and achieve high-order hp-convergence for regression problems. Section 5.1 poses a new architecture designed to admit closed form expressions for quadrature when discretizing variational PDEs. The presentation that follows however holds for an arbitrary POU architecture.

2.3. Whitney forms

Whitney k-forms are finite-dimensional differential k-forms on a simplicial complex K [48]. More precisely, the Whitney 0-form associated with the 0-simplex (vertex) $\boldsymbol{v}_i$ is the barycentric function λ_i. For $k > 0$, the Whitney k-form corresponding to the k-simplex with vertices $\boldsymbol{v}_{j_0}, \boldsymbol{v}_{j_1}, \cdots, \boldsymbol{v}_{j_k}$ is given by

$$\mathcal{W}_{j_0 \cdots j_k} = k! \sum_{i=0}^{k} (-1)^i \lambda_{j_i} \mathsf{d}\lambda_{j_0} \wedge \cdots \widehat{\mathsf{d}\lambda_{j_i}} \wedge \cdots \wedge \mathsf{d}\lambda_{j_k}, \tag{13}$$

where $\widehat{\mathsf{d}\lambda_{j_i}}$ means that the term is omitted. In total, there are $\begin{pmatrix} N^0 \\ k+1 \end{pmatrix}$ Whitney k-forms.

In 3D, the exterior derivative acting on 0-forms is $\mathsf{d} = \nabla$. The Whitney 1-form, for an edge between vertices $\boldsymbol{v}_i$ and $\boldsymbol{v}_j$ therefore reduces to

$$\mathcal{W}_{ij} = \lambda_i \nabla \lambda_j - \lambda_j \nabla \lambda_i. \tag{14}$$

Moreover, for a face with vertices $\boldsymbol{v}_i$, $\boldsymbol{v}_j$, and $\boldsymbol{v}_k$, the Whitney 2-form is

$$\mathcal{W}_{ijk} = (\lambda_i \nabla \lambda_j \times \nabla \lambda_k) - (\lambda_j \nabla \lambda_i \times \nabla \lambda_k) + (\lambda_k \nabla \lambda_i \times \nabla \lambda_j). \tag{15}$$

Note that $\mathcal{W}_{ij} = 0$ if $i = j$ and $\mathcal{W}_{ijk} = 0$ if i, j, and k are not distinct.

Since Whitney forms are constructed from barycentric coordinates, which form a POU, we have the following property of the Whitney forms.

Property 2.1. *Whitney forms form a partition of unity. In particular, the span of Whitney k-forms, denoted by W^k, contains all constant k-forms.*

Finally, the Whitney forms also have the following exactness property with respect to the exterior derivative.

Property 2.2. *Consider the sequence*

$$W^0 \xrightarrow{\mathsf{d}_0} W^1 \xrightarrow{\mathsf{d}_1} \cdots \xrightarrow{\mathsf{d}_{d-1}} W^d. \tag{16}$$

If Ω has trivial homology, then (16) is an exact sequence, i.e., $\mathrm{Null}(\mathsf{d}_k) = \mathrm{Range}(\mathsf{d}_{k-1})$ *for* $k > 1$.

Many generalizations of Whitney forms have been developed in the literature. For example, when Ω is a polytope, generalized Whitney forms constructed from generalized barycentric coordinates have been studied, see [28]. In this case, since the generalized barycentric coordinates might not be linear polynomials, we lose certain properties of the Whitney forms. However, Property 2.1 and 2.2 still hold in general. We refer readers to [28] for more discussions.

3. Whitney forms from POUs

We will now construct Whitney forms from POUs rather than barycentric coordinates, obtaining a generalization which admits interpretation as a control volume analysis and preserves conservation and exact sequence structure. For exposition, we restrict our presentation to $\Omega \subset \mathbb{R}^3$.

We start from a parameterized POU $\left\{\phi_i(\boldsymbol{x};\xi)\right\}_{i=1}^{N^0}$ where ξ denotes trainable parameters in the POU architecture. Without confusion, we will drop the dependence on ξ, but it is understood that ξ will be calibrated to adapt partitions to data. Mimicking the standard Whitney form construction, we adopt these POUs as Whitney 0-forms. Namely, we define POU Whitney 0-forms as

$$\psi_i = \phi_i, \quad i = 1, \cdots, N^0 \tag{17}$$

and introduce the corresponding finite-dimensional function space

$$V^0 = \left\{ \sum_i c_i \psi_i(x) \mid c_i \in \mathbb{R} \right\} \tag{18}$$

of dimension N^0.

By application of Stokes's Theorem (5), for $\boldsymbol{u} \in H_0(\mathrm{div}, \Omega)$, we derive

$$\int_\Omega \psi_i \nabla \cdot \boldsymbol{u} = -\int_\Omega \nabla \phi_i \cdot \boldsymbol{u},$$

and note that

$$\begin{aligned}
&& \mathrm{grad}\, \psi_i &= \nabla \phi_i \\
&(\sum_j \phi_j = 1) & &= \left(\sum_j \phi_j\right) \nabla \phi_j \\
&& &= \phi_i \nabla \phi_i + \sum_{j \neq i} \phi_j \nabla \phi_i \\
&(\sum_j \nabla \phi_j = 0) & &= \phi_i \left(-\sum_{j \neq i} \nabla \phi_j \right) + \sum_{j \neq i} \phi_j \nabla \phi_i \\
&& &= (-1) \sum_{j \neq i} (\phi_i \nabla \phi_j - \phi_j \nabla \phi_i)
\end{aligned}$$

We obtain the definition of ψ_{ij}

$$\int_\Omega \psi_i \nabla \cdot \boldsymbol{u} = \sum_{j \neq i} \int_\Omega (\phi_i \nabla \phi_j - \phi_j \nabla \phi_i) \cdot \boldsymbol{u} =: \sum_{j \neq i} \int_\Omega \psi_{ij} \cdot \boldsymbol{u}, \tag{19}$$

where

$$\psi_{ij} = \phi_i \nabla \phi_j - \phi_j \nabla \phi_i. \tag{20}$$

The definition of ψ_{ij} resembles the classical Whitney 1-form (14), and we therefore refer to ψ_{ij} (20) as the POU Whitney 1-form between a pair of index i and j.

We note the correspondence between (19) and a traditional discrete divergence operator. Taking ψ_i as an indicator function over a compact domain Ω, and taking $\boldsymbol{u} \in H_0(\mathrm{div}, \Omega)$, due to the antisymmetry properties of (20), we obtain the Gauss divergence theorem

$$\int_\Omega \nabla \cdot \boldsymbol{u} = \sum_i \int_\Omega \psi_i \nabla \cdot \boldsymbol{u} = \sum_{ij} \int_\Omega \psi_{ij} \cdot \boldsymbol{u} = \frac{1}{2}\left(\sum_{ij} \int_\Omega \psi_{ij} \cdot \boldsymbol{u} - \sum_{ji} \int_\Omega \psi_{ji} \cdot \boldsymbol{u} \right) = 0.$$

ψ_i and ψ_{ij} thus correspond to volume and area *densities*, with ψ_{ij} defining the boundary shared by ψ_i and ψ_j. These admit interpretation as diffuse representations of traditional mesh entities (cells/faces) in which indicator functions defined on cells have been mollified, with corresponding oriented areas defined consistently with the Leibniz rule.

We again define the corresponding finite-dimensional function space as

$$V^1 = \left\{ \sum_{ij} c_{ij} \psi_{ij}(x) \mid c_{ij} \in \mathbb{R} \right\},$$

where $\dim V^1 = N^1 = \binom{N^0}{2}$ and, naturally, our derivation above shows that $\operatorname{grad} V^0 \subseteq V^1$ by design.

Proceeding along the de Rham complex, we similarly apply the Kelvin-Stokes theorem for $\boldsymbol{u} \in H_0(\mathrm{curl}, \Omega)$ to obtain

$$\int_\Omega \psi_{ij} \nabla \times \boldsymbol{u} = \int_\Omega \nabla \times \psi_{ij} \cdot \boldsymbol{u}.$$

In a similar derivation, we obtain for $i \neq j$,

$$\begin{aligned}
& \operatorname{curl} \psi_{ij} = \nabla \times \psi_{ij} = \nabla \times (\phi_i \nabla \phi_j - \phi_j \nabla \phi_i) = 2(\nabla \phi_i \times \nabla \phi_j) \\
(\textstyle\sum_k \phi_k = 1) \qquad & = 2\left(\sum_k \phi_k \right)(\nabla \phi_i \times \nabla \phi_j) \\
& = 2\left\{ \phi_i(\nabla \phi_i \times \nabla \phi_j) + \phi_j(\nabla \phi_i \times \nabla \phi_j) + \sum_{k \neq i,j} \phi_k(\nabla \phi_i \times \nabla \phi_j) \right\} \\
(\textstyle\sum_k \nabla \phi_k = 0) \qquad & = 2\left\{ \phi_i(-\sum_{k \neq i} \nabla \phi_k \times \nabla \phi_j) + \phi_j(\nabla \phi_i \times -\sum_{k \neq j} \nabla \phi_k) + \sum_{k \neq i,j} \phi_k(\nabla \phi_i \times \nabla \phi_j) \right\} \\
& = 2 \sum_{k \neq i,j} (\phi_i \nabla \phi_j \times \nabla \phi_k - \phi_j \nabla \phi_i \times \nabla \phi_k + \phi_k \nabla \phi_i \times \nabla \phi_j).
\end{aligned}$$

We obtain

$$\int_\Omega \psi_{ij} \nabla \times \boldsymbol{u} = 2 \sum_{k \neq i,j} \int_\Omega \psi_{ijk} \cdot \boldsymbol{u}, \tag{21}$$

where, following (15), the POU Whitney 2-form ψ_{ijk} is defined as

$$\psi_{ijk} := (\phi_i \nabla \phi_j \times \nabla \phi_k) - (\phi_j \nabla \phi_i \times \nabla \phi_k) + (\phi_k \nabla \phi_i \times \nabla \phi_j). \tag{22}$$

Note that $\psi_{ijk} = -\psi_{\sigma(ijk)}$ where $\sigma(ijk)$ is an odd permutation of (i, j, k) and $\psi_{ijk} = 0$ if i, j, k are not distinct. Therefore, ψ_{ijk} correspond to diffuse generalizations of the circulations $\int \boldsymbol{u} \cdot d\boldsymbol{l}$ along an edge, providing the same conservation of circulation principle as the classical Kelvin-Stokes theorem. We introduce the corresponding finite-dimensional function space as follows,

$$V^2 = \left\{ \sum_{ijk} c_{ijk} \psi_{ijk}(x) \mid c_{ijk} \in \mathbb{R} \right\},$$

where $\dim V^2 = N^2 = \binom{N^0}{3}$. Similarly, our derivation implies that $\operatorname{curl} V^1 \subseteq V^2$.

Finally, the POU Whitney 3-form can be derived in the same way. Starting from the Stokes' Theorem (5) for div, i.e., for $u \in H_0(\mathrm{grad}, \Omega)$,

$$\int_\Omega \psi_{ijk} \cdot \nabla u = -\int_\Omega \nabla \cdot \psi_{ijk}\, u,$$

and the fact that, for $i \neq j \neq k$

$$\operatorname{div} \psi_{ijk} = \nabla \cdot \psi_{ijk} = 6\left(\nabla \phi_i \times \nabla \phi_j \cdot \nabla \phi_k \right)$$

$$
\begin{aligned}
(\sum_l \phi_l = 1) \qquad &= 6\left(\sum_l \phi_l\right)\left(\nabla\phi_i \times \nabla\phi_j \cdot \nabla\phi_k\right) \\
&= 6\{\left[\sum_{l\neq i,j,k} \phi_l\left(\nabla\phi_i \times \nabla\phi_j \cdot \nabla\phi_k\right)\right] + \phi_i\left(\nabla\phi_i \times \nabla\phi_j \cdot \nabla\phi_k\right) \\
&\quad + \phi_j\left(\nabla\phi_i \times \nabla\phi_j \cdot \nabla\phi_k\right) + \phi_k\left(\nabla\phi_i \times \nabla\phi_j \cdot \nabla\phi_k\right)\} \\
(\sum_l \nabla\phi_l = 0) \qquad &= 6\{\left[\sum_{l\neq i,j,k} \phi_l\left(\nabla\phi_i \times \nabla\phi_j \cdot \nabla\phi_k\right)\right] + \phi_i\left(-\sum_{l\neq i} \nabla\phi_l \times \nabla\phi_j \cdot \nabla\phi_k\right) \\
&\quad + \phi_j\left(\nabla\phi_i \times -\sum_{l\neq j} \nabla\phi_l \cdot \nabla\phi_k\right) + \phi_k\left(\nabla\phi_i \times \nabla\phi_j \cdot -\sum_{l\neq k} \nabla\phi_l\right)\} \\
&= -6\sum_{l\neq i,j,k}(\phi_i\nabla\phi_j \times \nabla\phi_k \cdot \nabla\phi_l - \phi_j\nabla\phi_i \times \nabla\phi_k \cdot \nabla\phi_l \\
&\quad + \phi_k\nabla\phi_i \times \nabla\phi_j \cdot \nabla\phi_l - \phi_l\nabla\phi_i \times \nabla\phi_j \cdot \nabla\phi_k),
\end{aligned}
$$

we have

$$
\int_\Omega \psi_{ijk} \cdot \nabla u = 6\sum_{l\neq i,j,k}\int_\Omega \psi_{ijkl} u, \tag{23}
$$

where we define the POU Whitney 3-form

$$
\psi_{ijkl} := \phi_i\nabla\phi_j \times \nabla\phi_k \cdot \nabla\phi_l - \phi_j\nabla\phi_i \times \nabla\phi_k \cdot \nabla\phi_l + \phi_k\nabla\phi_i \times \nabla\phi_j \cdot \nabla\phi_l - \phi_l\nabla\phi_i \times \nabla\phi_j \cdot \nabla\phi_k.
$$

Again, we have $\psi_{ijkl} = -\psi_{\sigma(ijkl)}$ where $\sigma(ijkl)$ is an odd permutation of (i,j,k,l) and $\psi_{ijkl} = 0$ if i,j,k,l are not distinct. The corresponding finite-dimensional function space is

$$
V^3 = \left\{\sum_{ijkl} c_{ijkl}\psi_{ijkl}(x) \mid c_{ijkl} \in \mathbb{R}\right\},
$$

where $\dim V^3 = N^3 = \binom{N^0}{4}$. Finally, it is clear that $\operatorname{div} V^2 \subseteq V^3$.

We may use the POU Whitney forms to define the following set of discrete exterior derivatives generalizing the familiar vector calculus div/grad/curl, which may be used to discretize PDEs.

$$
DIV(U)_i := \sum_{j\neq i} U_{ij}, \quad U_{ij} = \int_\Omega \psi_{ij} \cdot \boldsymbol{u}, \ \forall \boldsymbol{u} \in H_0(\mathrm{div}, \Omega) \tag{24}
$$

$$
CURL(U)_{ij} := \sum_{k\neq i,j} U_{ijk}, \quad U_{ijk} = 2\int_\Omega \psi_{ijk} \cdot \boldsymbol{u}, \ \forall \boldsymbol{u} \in H_0(\mathrm{curl}, \Omega) \tag{25}
$$

$$
GRAD(U)_{ijk} := \sum_{l\neq i,j,k} U_{ijkl}, \quad U_{ijkl} = 6\int_\Omega \psi_{ijkl}\, u, \ \forall u \in H_0(\mathrm{grad}, \Omega). \tag{26}
$$

As discussed previously, local conservation principles follow from the anti-symmetry in the summands of (24), (25), and (26). We next demonstrate that the exact sequence property holds at a discrete level as well:

$$
\begin{aligned}
DIV(CURL(U))_i &= \sum_{j\neq i} CURL(U)_{ij} && \textit{(definition of } DIV\textit{)} \\
&= \sum_{j\neq i}\sum_{k\neq i,j} U_{ijk} && \textit{definition of } CURL \\
&= \sum_{j\neq i}\sum_{k\neq i,j} 2\int_\Omega \psi_{ijk} \cdot \boldsymbol{u} && \textit{definition of } U \\
&= \sum_{j\neq i}\int_\Omega \psi_{ij}\nabla \times \boldsymbol{u} && \textit{Leibniz rule} \\
&= \int_\Omega \psi_i \nabla \cdot \nabla \times \boldsymbol{u} && \textit{Leibniz rule} \\
&= 0,
\end{aligned}
$$

and similarly we obtain

$$\begin{aligned} CURL(GRAD(U))_{ij} &= \sum_{k\neq i,j} GRAD(U)_{ijk} = \sum_{k\neq i,j}\sum_{l\neq i,j,k} U_{ijkl} \\ &= \sum_{k\neq i,j}\sum_{l\neq i,j,k} 6\int_\Omega \psi_{ijkl} u = \sum_{k\neq i,j}\int_\Omega \psi_{ijk}\cdot\nabla u \\ &= \frac{1}{2}\int_\Omega \psi_{ij}\nabla\times\nabla u = 0. \end{aligned}$$

In fact, the above properties follow directly if we notice that the discrete differential operators (24)–(26) are closely related to the *coboundary operators* used widely in the *combinatorial graph exterior calculus*; which we briefly recall here. Consider a complete graph $\mathcal{G} = (\mathcal{V},\mathcal{E})$ with the vertex set $\mathcal{V} = \{1,2,\cdots,N^0\}$ and edge set $\mathcal{E}$, embedded in $\mathbb{R}^3$. Let C_k be the set of k-chains (a linear combination of $(k+1)$-cliques on the graph) and C^k be the set of cochains. Note that C^k consists of linear functionals acting on C_k. In the graph setting, we define the coboundary operator $\delta_k : C^k \to C^{k+1}$ as the *signed incidence matrix* between the k-cliques and $(k+1)$-cliques (see e.g., [29]). In 3D, δ_0 is the edge-vertex signed incidence matrix, δ_1 is the triangle-edge signed incidence matrix, and δ_2 is the tetrahedron-triangle signed incidence matrix. We should understand triangles and tetrahedrons as the 3-cliques and 4-cliques on the complete graph $\mathcal{G}$, respectively. Thus, from the definitions of the coboundary operators [29] and the definitions (24)–(26), and aligning the orientations of cliques and ordering of the indices in the summations (24)–(26), we may identify the discrete exterior derivatives with the adjoints of the graph operators.

$$DIV = \delta_0^T, \quad CURL = 2\delta_1^T, \quad GRAD = 6\delta_2^T. \tag{27}$$

Moreover,

$$\begin{aligned} DIV\,CURL &= \delta_0^T\, 2\,\delta_1^T = 2\left(\delta_1\delta_0\right)^T = 0 \\ CURL\,GRAD &= 2\,\delta_1^T\, 6\,\delta_2^T = 12\left(\delta_2\delta_1\right)^T = 0 \end{aligned}$$

In the remainder of this paper, we employ the FEEC framework to develop Whitney forms as a finite-element basis. Note that the finite dimensional function spaces V^0, V^1, V^2, and V^3 were constructed to satisfy the complex

$$V^3 \xleftarrow[-\mathrm{div}]{} V^2 \xleftarrow[\mathrm{curl}]{} V^1 \xleftarrow[-\mathrm{grad}]{} V^0, \tag{28}$$

which is a subcomplex of the dual de Rham complex (7). This suggests that, as in the FEEC setting, we should use the POU Whitney forms to approximate the functions from the dual point of view and consider the following POU complex

$$V^3 \xrightarrow{\mathrm{grad}} V^2 \xrightarrow{\mathrm{curl}} V^1 \xrightarrow{\mathrm{div}} V^0, \tag{29}$$

which is a subcomplex of the primal de Rham complex (4). Therefore, to discretize the divergence operator, we consider $\boldsymbol{u}\in V^1$ and $q \in V^0$ and

$$(\nabla\cdot\boldsymbol{u}, q) = (\boldsymbol{u}, -\nabla q) = \left(\sum_{ab} u_{ab}\psi_{ab}, \sum_i q_i(-\nabla)\psi_i\right) =: \mathsf{q}^T\,\mathsf{DIV}\,\mathsf{u},$$

where u and q are vectors expansions of $\boldsymbol{u}$ and q in a basis for V^1 and V^0, respectively, and the matrix DIV is defined as

$$(\mathsf{DIV})_{i,(ab)} = (\psi_{ab}, -\nabla\psi_i). \tag{30}$$

Similarly, to discretize the curl operator we consider $\boldsymbol{u}\in V^2$ and $\boldsymbol{v}\in V^1$, and then

$$(\nabla\times\boldsymbol{u},\boldsymbol{v}) = (\boldsymbol{u},\nabla\times\boldsymbol{v}) =: \mathsf{v}^T\,\mathsf{CURL}\,\mathsf{u},$$

where u and v are vector representations of $\boldsymbol{u}$ and $\boldsymbol{v}$, respectively, and the matrix CURL is defined as

$$(\mathsf{CURL})_{(ij),(abc)} = (\psi_{abc}, \nabla\times\psi_{ij}). \tag{31}$$

Finally, for the gradient operator, we use $u\in V^3$ and $\boldsymbol{v}\in V^2$, and

$$(\nabla u, \boldsymbol{v}) = (u, -\nabla\cdot\boldsymbol{v}) =: \mathsf{v}^T\,\mathsf{GRAD}\,\mathsf{u},$$

where u and v are vector representation of u and $\boldsymbol{v}$, respectively, and the matrix GRAD is defined as

$$(\mathsf{GRAD})_{(ijk),(abcd)} = (\psi_{abcd}, -\nabla\cdot\psi_{ijk}). \tag{32}$$

Similar to the traditional low-order FEEC/DEC setting [61], the FEEC differential operators (30)–(32) are closely related to the DEC differential operators (24)–(26). Based on the properties $-\nabla\psi_i = \sum_{j\neq i}\psi_{ij}$, $\nabla\times\psi_{ij} = 2\sum_{k\neq i,j}\psi_{ijk}$, and $-\nabla\cdot\psi_{ij} = (-6)\sum_{\ell\neq i,j,k}\psi_{ijkl}$, we have

$$\begin{aligned}
(\mathsf{DIV})_{i,(ab)} &= (\psi_{ab}, -\nabla\psi_i) = \sum_{j\neq i}(\psi_{ab}, \psi_{ij}),\\
(\mathsf{CURL})_{(ij),(abc)} &= (\psi_{abc}, \nabla\times\psi_{ij}) = 2\sum_{k\neq i,j}(\psi_{abc}, \psi_{ijk}),\\
(\mathsf{GRAD})_{(ijk),(abcd)} &= (\psi_{abcd}, -\nabla\cdot\psi_{ijk}) = 6\sum_{l\neq i,j,k}(\psi_{abcd}, \psi_{ijkl}),
\end{aligned}$$

which provides the identities

$$\mathsf{DIV} = DIV\,\mathbf{M}_1, \quad \mathsf{CURL} = CURL\,\mathbf{M}_2, \quad \mathsf{GRAD} = GRAD\,\mathbf{M}_3,$$

where $(\mathbf{M}_1)_{(ij),(ab)} = (\psi_{ab}, \psi_{ij})$, $(\mathbf{M}_2)_{(ijk),(abc)} = (\psi_{abc}, \psi_{ijk})$, and $(\mathbf{M}_3)_{(ijkl),(abcd)} = (\psi_{abcd}, \psi_{ijkl})$ are mass matrices associated with the finite element spaces V^1, V^2, and V^3, respectively. Furthermore, using (27), we have

$$\mathsf{DIV} = \delta_0^T\mathbf{M}_1, \quad \mathsf{CURL} = 2\,\delta_1^T\mathbf{M}_2, \quad \mathsf{GRAD} = 6\,\delta_2^T\mathbf{M}_3. \tag{33}$$

We remark that the scalings 2 and 6 above do not effect the finite dimensional spaces spanned by the POU Whitney forms, and we therefore may drop them without loss of generality in the variational setting.

Equation (33) is the cornerstone of this work; it reveals an intimate connection between the graph exterior calculus we have used as a basis for learning physics in [71] and the finite element exterior calculus. Whereas in [72] data was used to associate a metric with a given graph, (33) shows that the graph itself and the associated metric information may be inferred directly from the Whitney forms. The entries of the matrices $\mathbf{M}_k$ encode graph sparsity by weighting boundary interactions between partitions, and therefore the geometric problem of identifying control volumes is equivalent to a graph discovery problem.

4. Generalizations to high-dimensional manifolds

In this section, we generalize the construction of POU Whitney forms to high-dimensional manifolds. Similar to the prior case, we define the 0-th order Whitney forms as a given POU on the manifold, and then define higher-order Whitney forms by induction using the Leibniz rule and properties of POUs to finally arrive at an equivalence between graph coboundary operators, discrete exterior derivatives, and FEEC derivatives. The results of this section directly parallel those of Section 3. Again, let

$$\psi_i = \phi_i, \quad i = 1, \cdots, N^0 \tag{34}$$

the same as in (17), but now $\psi_i : \mathcal{M} \to [0,1]$ for a smooth manifold $\mathcal{M} \subseteq \mathbb{R}^d$. Paralleling (13), we define higher-order POU Whitney k-forms as follows,

$$\psi^k_{j_0\cdots j_k} = \sum_{i=0}^{k}(-1)^i\phi_{j_i}\,\mathsf{d}\phi_{j_0}\wedge\cdots\wedge\widehat{\mathsf{d}\phi_{j_i}}\wedge\cdots\wedge\mathsf{d}\phi_{j_k}. \tag{35}$$

The exterior derivative of the k-th order Whitney form is given by

$$\begin{aligned}
\mathsf{d}\psi^k_{j_0\cdots j_k} &= \mathsf{d}\left(k!\sum_{i=0}^{k}(-1)^i\phi_{j_i}\mathsf{d}\phi_{j_0}\wedge\cdots\wedge\widehat{\mathsf{d}\phi_{j_i}}\wedge\cdots\wedge\mathsf{d}\phi_{j_k}\right)\\
&= k!\left(\sum_{i=0}^{k}(-1)^i\mathsf{d}\phi_{j_i}\wedge\mathsf{d}\phi_{j_0}\wedge\cdots\wedge\widehat{\mathsf{d}\phi_{j_i}}\wedge\cdots\wedge\mathsf{d}\phi_{j_k}\right)\\
&= k!\left(\sum_{i=0}^{k}(-1)^i(-1)^i\mathsf{d}\phi_{j_0}\wedge\cdots\wedge\mathsf{d}\phi_{j_i}\wedge\cdots\wedge\mathsf{d}\phi_{j_k}\right)\\
&= (k+1)!\left(\mathsf{d}\phi_{j_0}\wedge\cdots\wedge\mathsf{d}\phi_{j_k}\right).
\end{aligned}$$

Applying the POU properties from (8) and (9), we obtain

$$\begin{aligned}
&\mathsf{d}\phi_{j_0}\wedge\cdots\wedge\mathsf{d}\phi_{j_k}\\
(\textstyle\sum_{j_{k+1}}\phi_{j_{k+1}}=1) \qquad &= \left(\sum_{j_{k+1}}\phi_{j_{k+1}}\right)\left(\mathsf{d}\phi_{j_0}\wedge\cdots\wedge\mathsf{d}\phi_{j_k}\right)\\
&= \sum_{j_{k+1}\neq j_0,\cdots,j_k}\phi_{j_{k+1}}\left(\mathsf{d}\phi_{j_0}\wedge\cdots\wedge\mathsf{d}\phi_{j_k}\right) + \sum_{i=0}^{k}\phi_{j_i}\left(\mathsf{d}\phi_{j_0}\wedge\cdots\wedge\mathsf{d}\phi_{j_i}\wedge\cdots\wedge\mathsf{d}\phi_{j_k}\right)\\
(\textstyle\sum_{j_{k+1}}\mathsf{d}\phi_{j_{k+1}}=0) \qquad &= \sum_{j_{k+1}\neq j_0,\cdots,j_k}\phi_{j_{k+1}}\left(\mathsf{d}\phi_{j_0}\wedge\cdots\wedge\mathsf{d}\phi_{j_k}\right)
\end{aligned}$$

$$
\begin{aligned}
&+\sum_{i=0}^{k} \phi_{j_i}\left(\mathrm{d}\phi_{j_0}\wedge\cdots\wedge\left[-\sum_{j_{k+1}\neq j_i}\mathrm{d}\phi_{j_{k+1}}\right]\wedge\cdots\wedge\mathrm{d}\phi_{j_k}\right)\\
=&\sum_{j_{k+1}\neq j_0,\cdots,j_k}\phi_{j_{k+1}}\left(\mathrm{d}\phi_{j_0}\wedge\cdots\wedge\mathrm{d}\phi_{j_k}\right)\\
&-\sum_{i=0}^{k}\sum_{j_{k+1}\neq j_0,\cdots,j_k}\phi_{j_i}\left(\mathrm{d}\phi_{j_0}\wedge\cdots\wedge\mathrm{d}\phi_{j_{k+1}}\wedge\cdots\wedge\mathrm{d}\phi_{j_k}\right)\\
=&\sum_{j_{k+1}\neq j_0,\cdots,j_k}\phi_{j_{k+1}}\left(\mathrm{d}\phi_{j_0}\wedge\cdots\wedge\mathrm{d}\phi_{j_k}\right)\\
&-\sum_{i=0}^{k}\sum_{j_{k+1}\neq j_0,\cdots,j_k}(-1)^{k-i}\phi_{j_i}\left(\mathrm{d}\phi_{j_0}\wedge\cdots\wedge\widehat{\mathrm{d}\phi_{j_i}}\wedge\cdots\wedge\mathrm{d}\phi_{j_k}\wedge\mathrm{d}\phi_{j_{k+1}}\right)\\
=&(-1)^{k+1}\sum_{j_{k+1}\neq j_0,\cdots,j_k}\left(\sum_{i=0}^{k+1}(-1)^i\phi_{j_i}\mathrm{d}\phi_{j_0}\wedge\cdots\wedge\widehat{\mathrm{d}\phi_{j_i}}\wedge\cdots\wedge\mathrm{d}\phi_{j_{k+1}}\right).
\end{aligned}
$$

Here $\sum_{j_{k+1}\neq j_0,\cdots,j_k}\phi_{j_{k+1}}$ implies a summation over indices $j_{k+1}\in\{1,\dots,N^0\}$ which do not take the same value as any of the indices $j_0,\dots,j_k\in\{1,\dots,N^0\}$. Finally, we arrive at the following expression allowing construction of the POU Whitney $(k+1)$-forms from k-forms.

Property 4.1.

$$
\mathrm{d}\psi^k_{j_0\cdots j_k}=(-1)^{k+1}(k+1)!\sum_{j_{k+1}\neq j_0,\cdots,j_k}\psi^{k+1}_{j_0\cdots j_{k+1}}
$$

Proof. The result follows from the previous two equalities and the definition of the POU Whitney k-form (35)

$$
\mathrm{d}\psi^k_{j_0\cdots j_k}=(-1)^{k+1}(k+1)!\sum_{j_{k+1}\neq j_0,\cdots,j_k}\left(\sum_{i=0}^{k+1}(-1)^i\phi_{j_i}\mathrm{d}\phi_{j_0}\wedge\cdots\wedge\widehat{\mathrm{d}\phi_{j_i}}\wedge\cdots\wedge\mathrm{d}\phi_{j_{k+1}}\right).\quad\square
$$

One consequence of Property 4.1 is that, for any $u\in\Lambda^{d-k-1}$ satisfying $\operatorname{tr}u=0$, based on (6), we have

$$
\begin{aligned}
\int_\Omega \psi^k_{j_0\cdots j_k}\wedge\mathrm{d}u&=(-1)^{k+1}\int_\Omega \mathrm{d}\psi^k_{j_0\cdots j_k}\wedge u\\
&=(-1)^{k+1}\int_\Omega\left[(-1)^{k+1}(k+1)!\sum_{j_{k+1}\neq j_0,\cdots,j_k}\psi^{k+1}_{j_0\cdots j_{k+1}}\right]\wedge u\\
&=(k+1)!\sum_{j_{k+1}\neq j_0,\cdots,j_k}\int_\Omega\psi^{k+1}_{j_0\cdots j_{k+1}}\wedge u.
\end{aligned}
$$

The above formula naturally suggests the DEC discretization of differential operators

$$
D_{d-k-1}(U)_{j_0\cdots j_k}=\sum_{j_{k+1}\neq j_0,\cdots,j_k}U_{j_0\cdots j_{k+1}},\quad U_{j_0\cdots j_{k+1}}=(k+1)!\int_\Omega\psi^{k+1}_{j_0\cdots j_{k+1}}\wedge\boldsymbol{u},\qquad\forall\boldsymbol{u}\in\Lambda^{d-k-1}. \tag{36}
$$

This operator is locally conservative via the permutation property of $\psi^k_{j_0\cdots j_k}$, and maintains the discrete exact sequence property

$$
\begin{aligned}
D_{d-k-1}(D_{d-k-2}(U))_{j_0\cdots j_k}&=\sum_{j_{k+1}\neq j_0,\cdots,j_k}D(U)_{j_0\cdots j_{k+1}}=\sum_{j_{k+1}\neq j_0,\cdots,j_k}\sum_{j_{k+2}\neq j_0,\cdots,j_{k+1}}U_{j_0\cdots j_{k+2}}\\
&=\sum_{j_{k+1}\neq j_0,\cdots,j_k}\sum_{j_{k+2}\neq j_0,\cdots,j_{k+1}}(k+2)!\int_\Omega\psi^{k+2}_{j_0\cdots j_{k+2}}\wedge\boldsymbol{u}\\
&=\sum_{j_{k+1}\neq j_0,\cdots,j_k}\int_\Omega\psi^{k+1}_{j_0\cdots j_{k+1}}\wedge\mathrm{d}\boldsymbol{u}\\
&=\frac{1}{(k+1)!}\int_\Omega\psi^k_{j_0\cdots j_k}\wedge\mathrm{d}(\mathrm{d}\boldsymbol{u})=0.
\end{aligned}
$$

Similar to the 3D case, by introducing a complete graph $\mathcal{G}=(\mathcal{V},\mathcal{E})$ with the vertex set $\mathcal{V}=\{1,2,\cdots,N^0\}$ and the edge set $\mathcal{E}$, which consists all possible edges between any two vertices, and, again, aligning the orientation of the cliques and the ordering used in the summation definition (36), the DEC differential operators D are closely related to the coboundary operators δ_k, i.e.,

$$D_{d-k-1} = (k+1)!\,\delta_k^T, \tag{37}$$

where δ_k are the signed incidence matrices between k-cliques and $(k+1)$-cliques.

This reinterpretation provides the simpler alternative proof of the exact sequence property.

$$D_{d-k-1}\,D_{d-k-2} = (k+1)!\,\delta_k^T\,(k+2)!\,\delta_{k+1}^T = (k+1)!(k+2)!\,\left(\delta_{k+1}\,\delta_k\right)^T = 0.$$

Following the 3D case, we can define corresponding finite-dimensional function spaces of k-forms as

$$V^k = \left\{ \sum_{j_0,\cdots,j_k} c_{j_0\cdots j_k}\psi^k_{j_0\cdots j_k} | c_{j_0\cdots j_k} \in \mathbb{R} \right\}$$

where $\dim V^k = \binom{N^0}{k+1}$, allowing recasting of the discrete differential operators into the FEEC setting. Again, we approximate functions from the dual point of view and obtain the dual de Rham complex

$$V^0 \xleftarrow[\mathsf{d}]{} \cdots \xleftarrow[\mathsf{d}]{} V^{d-1} \xleftarrow[\mathsf{d}]{} V^d.$$

For $u \in V^{k+1}$ and $v \in V^k$, let u and v be vector representations corresponding to basis expansions in V^{k+1} and V^k, and we obtain

$$(\mathsf{d}u, v) = (-1)^{k+1}(u, \mathsf{d}v) =: \mathsf{v}^T\,\mathsf{D}_{d-k-1}\,\mathsf{u}$$

where D_{d-k-1} is the FEEC discretization of the differential operator d defined as follows,

$$(\mathsf{D}_{d-k-1})_{(i_0\cdots i_k),(j_0\cdots j_{k+1})} = (\psi^{k+1}_{j_0\cdots j_{k+1}}, (-1)^{k+1}\mathsf{d}\psi^k_{i_0\cdots i_k}).$$

By Property 4.1, the FEEC differential operator D_{d-k-1}, the DEC differential operator D_{d-k-1}, and the coboundary operator δ_k (i.e., the signed incidence matrix on the complete graph $\mathcal{G}$), are equivalent in the following sense

$$\mathsf{D}_{d-k-1} = D_{d-k-1}\mathbf{M}_{k+1} = (k+1)!\,\delta_k^T\mathbf{M}_{k+1},$$

where $\mathbf{M}_{k+1}$ is the mass matrix of the finite element space V^{k+1}.

5. Data-driven Whitney forms for learning physics

To infer Whitney forms and physics from data, we will solve the following equality constrained quadratic program

$$\min_{\theta} ||u - u_{data}||^2 \tag{38}$$

$$\text{such that} \quad \mathcal{L}_\theta[u] = f, \tag{39}$$

where θ denotes parameters associated with learning both Whitney forms as well as the underlying physics. In [71], we introduced a complete stability theory for models of the form

$$\mathcal{L}_\theta[u]\Delta_k u + \mathcal{N}[\mathsf{d}u], \tag{40}$$

i.e., nonlinear perturbations of the k^{th} order Hodge Laplacian Δ_k by a nonlinear perturbation $\mathcal{N}[\mathsf{d}u]$ in conservation form and parameterized by a neural network. Following standard Lax-Milgram/fixed point analysis, this model class was shown to be well-posed given mild conditions on the nonlinearity. For simplicity in the current work, we will consider only the linear $\mathrm{div}-\mathrm{grad}$ system, but stress that the analysis from [71] holds and the FEEC extension may be applied to a more general class of problems.

Solving Equation (38) provides a set of partitions which optimally approximate data while satisfying a structure preserving control volume analysis. To accomplish this, in Section 5.1 we introduce a POU architecture that admits closed form expressions for quadrature, allowing us to consider variational models for $\mathcal{L}_\theta$ which are end-to-end differentiable, and therefore amenable to back propagation. Then in Section 5.2 we introduce a variational model for $\mathcal{L}_\theta$ and outline how to generalize the learning of metric information from DDEC to the FEEC setting. More details on the assembly process can be found in Appendix A.

5.1. Partition of unity construction

Use of the POU Whitney forms as FEEC shape functions requires their integration in the bilinear forms of the variational problem. Many architectures may be used to parameterize the POU; for example, popular classification architectures consisting of a multilayer perceptron composed with a softmax activation are natural choices [46]. However, such architectures require the use of inexact quadrature rules, introducing a variational crime and complicating numerical analysis. To avoid this, we introduce a novel architecture that admits exact expressions for integrals of shape functions and their derivatives. To do so, we construct POUs as convex combinations of tensor-product free-knot B-splines. B-splines admit a dual interpretation as ReLU networks [31,11], naturally form partitions of unity, and admit closed-form expressions for integrals depending only on the degree of the B-splines and knot locations [57,18]. While any degree of B-splines may be used, we choose to work with B1 splines. We sketch the construction of 0-forms and 1-forms in Figs. 1 and 2, respectively.

For the unit hypercube $\Omega = [0,1]^n$, we form n-dimensional B-splines via tensor products of 1D splines, with unique sets of knots along each dimension. Let $\boldsymbol{x} = (x_1, \dots, x_n) \in \Omega$. For each $d = 1, \dots, n$ let $\{\zeta_k(t) \mid k = 1, \dots, N_{x_d}\}$ be a set of 1-dimensional B1 spline basis functions with an associated set of spline nodes $T_d = \{t^d_{k_d} \mid k_d = 1, \dots, N_{x_d}\}$, where the knots are ordered so that $t^d_{k_d} < t^d_{k_d+1}$; they satisfy the Kronecker delta property ($\zeta_i(t^d_j) = \delta_{ij}$); and they conform to the boundary ($t^d_1 = 0$ and $t^d_{N_{x_d}} = 1$). By the linear reproduction property of B-splines we have the POU property: for all $x \in [0,1]$,

$$\sum_{k_d=1}^{N_{x_d}} \zeta_{k_d}(x) = 1.$$

We denote the tensor product of these splines $\zeta_{k_1 k_2 \dots k_n} : \Omega \to [0,1]$ as

$$\zeta_{k_1 k_2 \dots k_n}(\boldsymbol{x}) = \prod_{d=1}^{n} \zeta_{k_d}(x_d),$$

and denote the set of all spline knots $T \subset \Omega$ as

$$T = \left\{ \left(t^1_{k_1}, t^2_{k_2}, \dots, t^n_{k_n} \right) \mid t^d_{k_d} \in T_d,\ d = 1, \dots, n \right\},$$

denoting the total number of knots $N_{\text{knots}} = \prod_d N_{x_d}$. The POU property is closed under tensor products, i.e.,

$$\sum_{k_1=1}^{N_{x_1}} \cdots \sum_{k_n=1}^{N_{x_n}} \zeta_{k_1 \dots k_n}(\boldsymbol{x}) = \prod_{d=1}^{n} \left(\sum_{k_d=1}^{N_{x_d}} \zeta_{k_d}(x_d) \right) = 1.$$

We take a convex combination of the tensor product B-splines to arrive at a coarsened non-Cartesian partition of space. Define a trainable tensor of weights $\mathbf{W}$ of size $N_{x_1} \times \cdots \times N_{x_n} \times N^0$, with $\mathbf{W}$ constrained so that for all $k_1 \dots k_n$,

$$\mathbf{W}_{k_1 \dots k_n i} \geq 0 \qquad \text{and} \qquad \sum_i \mathbf{W}_{k_1 \dots k_n i} = 1.$$

We discuss choices of $\mathbf{W}$ in Section 5.3.2. With these convex combinations, and for $i = 1, \dots, N^0$, we finally define our coarse-scale POU functions $\phi_i : \Omega \to [0,1]$

$$\phi_i(\boldsymbol{x}) = \sum_{k_1=1}^{N_{x_1}} \cdots \sum_{k_n=1}^{N_{x_n}} \mathbf{W}_{k_1 \dots k_n i}\, \zeta_{k_1 \dots k_n}(\boldsymbol{x}).$$

The partition of unity property is closed under convex combinations; i.e., from the convex combination restriction on $\mathbf{W}$, we obtain for any point $\boldsymbol{x} \in \Omega$,

$$\sum_i \phi_i(\boldsymbol{x}) = \sum_i \sum_{k_1=1}^{N_{x_1}} \cdots \sum_{k_n=1}^{N_{x_n}} \mathbf{W}_{k_1 \dots k_n i} \zeta_{k_1 \dots k_n}(\boldsymbol{x}) = \sum_{k_1=1}^{N_{x_1}} \cdots \sum_{k_n=1}^{N_{x_n}} 1 \cdot \zeta_{k_1 \dots k_n}(\boldsymbol{x}) = 1. \tag{41}$$

Therefore, our construction satisfies the requirements of Equation (8). Following the definitions in Equation (18), we then define $\psi_i(\boldsymbol{x}) = \phi_i(\boldsymbol{x})$ and $V^0 = \operatorname{span}\{\psi_i \mid i = 1, \dots, N^0\}$, and assemble the spaces $V^1, V^2, \dots$. With careful construction, we can also define spaces with a trace of 0 along the boundary, e.g., $V^0_D = \operatorname{span}\{\psi_i \mid i = 1, \dots, N^0$ such that $\psi_i(x) = 0 \forall x \in \partial\Omega\}$, presented in the next subsection.

5.1.1. Boundary/interior POUs for Dirichlet conditions

When we later define the variational problem, we will need to perform a lift of the solution on the portion of the boundary $\Gamma_D \subset \partial\Omega$ associated with Dirichlet boundary conditions. Toward this end, we modify the convex combination tensor to provide: *interior partitions*, 0-forms whose restriction to the boundary is zero; and *boundary partitions*, 0-forms with a Kronecker delta property on Γ_D.

We restrict the convex combination tensor $\mathbf{W}$ into blocks corresponding to interior and boundary partitions. This restriction, and the corresponding block diagonal structure, may be expressed in terms of a matricization of $\mathbf{W}$, as defined in e.g., [40,41]. Matricization of a tensor $\mathbf{X} \in \mathbb{R}^{N_1 \times \cdots \times N_n}$ along index k reshapes $\mathbf{X}$ into a matrix $\mathbf{X}_{(k)}$ of size $\left(\prod_{\substack{d=1 \\ d \neq k}}^{n} N_d \right) \times N_k$. We restrict the form of $\mathbf{W}$ so that its matricization satisfies

$$\mathbf{W}_{(n+1)} = \begin{bmatrix} \mathbf{W}^{\text{int}}_{(n+1)} & 0 \\ 0 & \mathbf{W}^{\text{bdry}}_{(n+1)} \end{bmatrix}. \tag{42}$$

With N_{knots} total knots and $N_{\text{bc}} = \#\{T \cap \Gamma_D\}$ boundary knots, this restriction yields $\mathbf{W}^{\text{int}}_{(n+1)} \in \mathbb{R}^{(N_{\text{knots}} - N_{\text{bc}}) \times N^0_{\text{int}}}$ and $\mathbf{W}^{\text{bdry}}_{(n+1)} \in \mathbb{R}^{N_{\text{bc}} \times N^0_{\text{bdry}}}$. Fig. 1 provides an example POU consisting of three interior and two boundary POUs.

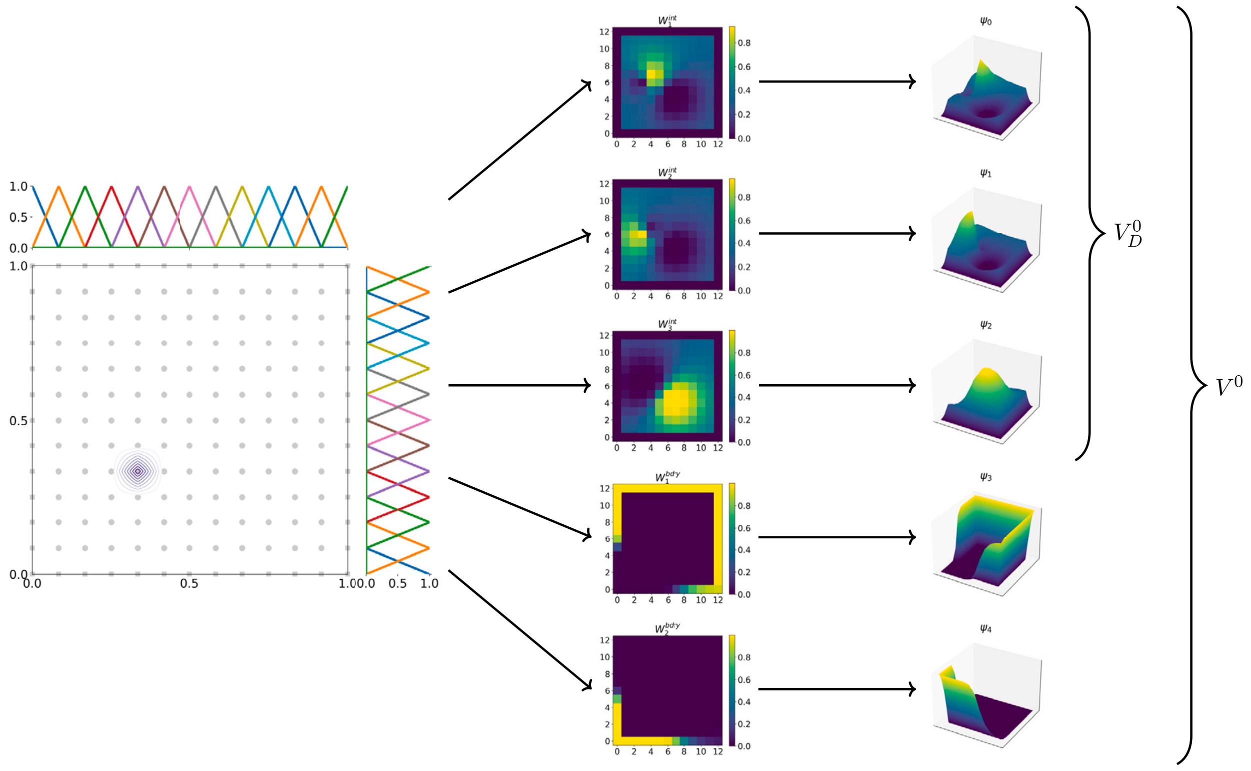


Fig. 1. Construction of 0-forms ψ_i. Convex combinations of tensor product free-knot B-splines form a basis for V^0 and V_D^0. *Left*: Interior spline nodes are marked with circles, while boundary nodes are squares. Contours of a single tensor product B-spline are plotted alongside knot locations. *Center*: Sparsity of convex combination tensor W for interior and boundary POUs. Partitioning is performed separately for interior and boundary knots, providing a basis with Kronecker delta property on the boundary suitable for imposing Dirichlet conditions. *Right:* The resulting basis functions parameterize the homogeneous component of the space (V_D^0) along with a basis for imposing Dirichlet conditions (V^0) via a lift (see Section 5.1.1). Quadrature may be performed by pulling back to the original tensor product space.

5.2. Variational model and graph network architecture

Let $\partial\Omega = \Gamma_N \cup \Gamma_D$, where Γ_D, Γ_N are disjoint. We consider as data solutions to the div-grad problem,

$$\begin{aligned} \mathbf{F} - \mu\nabla p &= 0 \\ \nabla\cdot\mathbf{F} &= f && \text{in } \Omega \\ p &= g_D && \text{on } \Gamma_D \\ \mathbf{F}\cdot\vec{n} &= g_N && \text{on } \Gamma_N, \end{aligned} \tag{43}$$

where $\mu \in L^2(\Omega)$ is a discontinuous diffusion coefficient.

To enforce boundary conditions, we split the pressure $p = p_0 + p_D$ into p_0 and p_D with $p_0 = 0$ and $p_D = g_D$ on Γ_D, and introduce the space

$$V_D^0 = \{\psi \in V^0 \mid \psi(x) = 0\ \forall x \in \Gamma_D\} = \operatorname{span}\left\{ \sum_{k_1=1}^{N_{x_1}} \cdots \sum_{k_n=1}^{N_{x_n}} W^{\text{int}}_{k_1\ldots k_n i}\, \zeta_{k_1\ldots k_n}(\boldsymbol{x}), \quad i = 1\ldots N^0_{\text{int}} \right\}.$$

We seek a solution $(p_0, \mathbf{F}) \in V_D^0 \times V^1$ that for all $(q_0, \mathbf{E}) \in V_D^0 \times V^1$ satisfies the mixed variational problem

$$\begin{aligned} (\mathbf{F},\mathbf{E}) - (\mathsf{d}^0 p_0, \mathbf{E}) &= (\mathsf{d}^0 p_D, \mathbf{E}) \\ -(\mathbf{F}, \mathsf{d}^0 q_0) &= (f, q_0) - (g_N, q_0)_{\Gamma_N}. \end{aligned} \tag{44}$$

Expanding the solution $(p_0, \mathbf{F})$ in terms of the Whitney forms

$$\begin{aligned} \mathbf{F} &= \sum_{i,j} \hat{F}_{ij}\psi_{ij} && \psi_{ij} \in V^1 \\ p_0 &= \sum_i \hat{p}_i \psi_i && \psi_i \in V_D^0, \end{aligned} \tag{45}$$

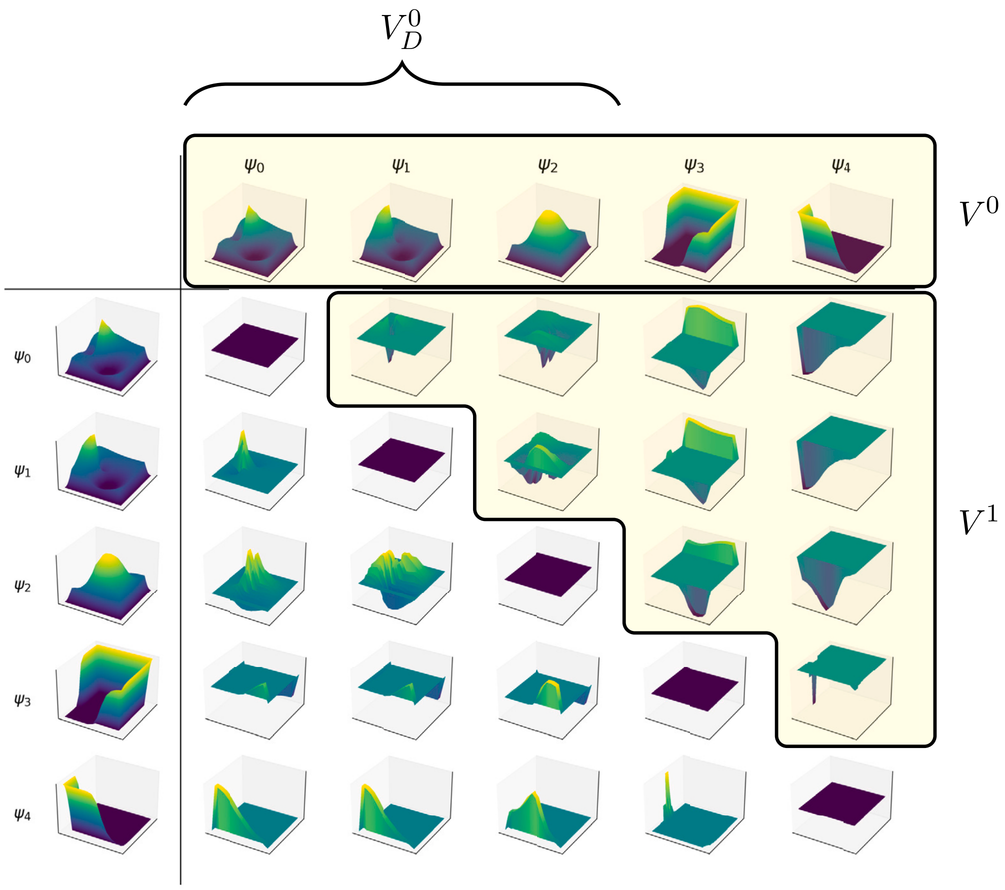


Fig. 2. Construction of 1-forms ψ_{ij} from pairs of 0-forms ψ_i and ψ_j constructed in Fig. 1. Following the definition $\psi_{ij} = \psi_i \nabla\psi_j - \psi_j \nabla\psi_i$, we observe the antisymmetry $\psi_{ij} = -\psi_{ji}$. X-components of the vector-valued 1-form are plotted.

we may recast Equation (44) in matrix form

$$\begin{bmatrix} \mathbf{M}_1 & -\mathsf{DIV}^T \\ -\mathsf{DIV} & \mathbf{0} \end{bmatrix} \begin{bmatrix} \hat{\mathbf{F}} \\ \hat{\mathbf{p}} \end{bmatrix} = \begin{bmatrix} \mathbf{b}_D \\ -\mathbf{b}_f - \mathbf{b}_N \end{bmatrix}, \tag{46}$$

where we have introduced $\hat{\cdot}$ to denote a vector of modal coefficients, and

$$\begin{aligned} b_{D(ij)} &= (\nabla p_D, \psi_{ij}) = \int_\Omega \nabla p_D \cdot \psi_{ij} \\ b_{f\,i} &= (f, \psi_i) = \int_\Omega f \psi_i \\ b_{N\,i} &= (g_N, \psi_i)_{\Gamma_N} = \int_{\Gamma_N} g_N \psi_i. \end{aligned} \tag{47}$$

Following [71], we introduce trainable symmetric positive semi-definite metric tensors $\mathbf{B}_0$, $\mathbf{D}_0$, $\mathbf{B}_1$, and $\mathbf{D}_1$ corresponding to Hodge star operators and parameterize d^0 and d^{0^T} as follows

$$\begin{aligned} \mathsf{d}^0 &= \mathbf{D}_1^{-1} \delta_0 \mathbf{D}_0 \\ \mathsf{d}^{0^T} &= \mathbf{B}_0^{-1} \delta_0^T \mathbf{B}_1 \end{aligned} \tag{48}$$

where δ_0 and δ_0^T are the related graph exterior calculus combinatorial operators on a complete graph with N^0 vertices. We thus have $\mathsf{DIV}^T = \mathbf{M}_1 \mathsf{d}^0$ and $\mathsf{DIV} = \mathsf{d}^{0^T} \mathbf{M}_1$, and finally arrive at our discrete model form

$$\begin{bmatrix} \mathbf{M}_1 & -\mathbf{M}_1 \mathbf{D}_1^{-1} \delta_0 \mathbf{D}_0 \\ -\mathbf{B}_0^{-1} \delta_0^T \mathbf{B}_1 \mathbf{M}_1 & \mathbf{0} \end{bmatrix} \begin{bmatrix} \hat{\mathbf{F}} \\ \hat{\mathbf{p}} \end{bmatrix} = \begin{bmatrix} \mathbf{b}_D \\ -\mathbf{b}_f - \mathbf{b}_N \end{bmatrix}. \tag{49}$$

Due to its block structure, assuming $\mathbf{M}_1$ is symmetric and positive definite, the solvability of (49) reduces to the invertibility of the Schur complement $\mathbf{S} = \mathbf{B}_0^{-1}\delta_0^T\mathbf{B}_1\mathbf{M}_1\mathbf{D}_1^{-1}\delta_0\mathbf{D}_0$ (note our definition of Schur complement here is slightly different from the convention by a negative sign). Note that the Schur complement $\mathbf{S}$ is basically a Hodge Laplacian defined in [71], which is invertible excluding the corresponding homology (see Theorem 3.5 in [71]). Therefore, we have the following theorem.

Theorem 5.1. *Assuming $\mathbf{B}_0$, $\mathbf{D}_0$, $\mathbf{B}_1$, $\mathbf{D}_1$, and $\mathbf{M}_1$ are symmetric positive definite, the mixed formulation (49) is solvable when excluding the corresponding homology of the Schur complement $\mathbf{S} = \mathbf{B}_0^{-1}\delta_0^T\mathbf{B}_1\mathbf{M}_1\mathbf{D}_1^{-1}\delta_0\mathbf{D}_0$.*

The underlying connection between FEEC and DEC is apparent if one chooses metric tensors satisfying $\mathbf{M}_1 \approx \mathbf{B}_1^{-1}\mathbf{D}_1$. In traditional FEM, this corresponds to performing a mass lumping [2,12,16]. If that approximation held, the linear system would reduce to

$$\begin{aligned} &\begin{bmatrix} \mathbf{M}_1 & -\mathbf{M}_1\mathbf{D}_1^{-1}\delta_0\mathbf{D}_0 \\ -\mathbf{B}_0^{-1}\delta_0^T\mathbf{B}_1\mathbf{M}_1 & \mathbf{0} \end{bmatrix} \begin{bmatrix} \hat{\mathbf{F}} \\ \hat{\mathbf{p}} \end{bmatrix} = \begin{bmatrix} \mathbf{b}_D \\ -\mathbf{b}_f - \mathbf{b}_N \end{bmatrix} \\ \iff \quad &\begin{bmatrix} \mathbf{B}_1^{-1} & \\ & \mathbf{B}_0^{-1} \end{bmatrix} \begin{bmatrix} \mathbf{I} & -\delta_0 \\ -\delta_0^T & \mathbf{0} \end{bmatrix} \begin{bmatrix} \mathbf{D}_1 & \\ & \mathbf{D}_0 \end{bmatrix} \begin{bmatrix} \hat{\mathbf{F}} \\ \hat{\mathbf{p}} \end{bmatrix} = \begin{bmatrix} \mathbf{b}_D \\ -\mathbf{b}_f - \mathbf{b}_N \end{bmatrix}. \end{aligned} \tag{50}$$

The assembly required to solve the linear system in Equation (49) is tedious but straightforward. Since our bases for V^0, V_D^0, and V^1 are each convex combinations of splines, the entries of $\mathbf{M}_1$ and DIV are given by analytic, closed-form expressions depending upon the spline knots T_d and convex combination tensor $\mathbf{W}$ (see Appendix A for example computation of mass matrices). To evaluate the forcing terms in Equation (47), the B1 spline interpolants of f, p_D, and g_N are used to perform exact quadrature.

We remark that analytic expressions for quadrature are crucial for the method to be computationally tractable. The scheme may be implemented in machine learning frameworks such as TensorFlow, PyTorch, or Jax which support automatic differentiation, so that the derivatives of Equation (49) are available. During training, the exact quadrature allows back propagation through the solution operator to evolve the control volumes. After training, the entries of the linear system in Equation (46) may be saved as a reduced order model which can be solved without reference to the underlying B-spline grid; the computational complexity associated with evaluating the B-spline space need only be incurred when post-processing the solution.

5.3. *Training*

Several variables in the above formulation are left to be optimized as a machine learning problem: the set of spline knots along each dimension T_d; the parameters associated with the convex combination tensor $\mathbf{W}$; and the exterior calculus metrics $\mathbf{D}_0$, $\mathbf{D}_1$, $\mathbf{B}_0$, $\mathbf{B}_1$.

5.3.1. *Constraints to ensure proper ordering of spline knots*

To integrate the Whitney forms, we require the knot ordering constraint ($t_k < t_{k+1}$) to be satisfied over the course of training. Rather than training knot locations t_k directly, we parameterize the distance between consecutive knots

$$d_k := t_k - t_{k-1} = \sigma(\tau_k),$$

where $\sigma(\cdot)$ is the sigmoid function and τ_k is a trainable parameter, ensuring $d_k > 0$ and $t_k < t_{k+1}$. The knot positions may be evaluated as $t_1 = 0$ and for $k > 1$,

$$t_k = \sum_{l=1}^{k} d_l.$$

The trainable variables τ_k are initialized by sampling a standard normal distribution.

5.3.2. *Constraints: convex combination tensor*

The convex combination tensor $\mathbf{W}$ must satisfy for all $k_1 \dots k_n$

$$\mathbf{W}_{k_1\dots k_n i} \geq 0 \quad \text{and} \quad \sum_i \mathbf{W}_{k_1\dots k_n i} = 1.$$

To enforce these constraints, we parameterize $\mathbf{W}$ by introducing a learnable tensor $\mathbf{V}$ and applying a softmax activation to the final index

$$\mathbf{W}_{k_1\dots k_n i} = \frac{\exp \mathbf{V}_{k_1\dots k_n i}}{\sum_i \exp \mathbf{V}_{k_1\dots k_n i}}, \tag{51}$$

where $\mathbf{V}_{k_1\dots k_n,i}$ denotes the weight associated with the knot located at $(t_{k_1}, \dots, t_{k_n})$.

In the case where $\Gamma_D \neq \emptyset$, we repeat this process for $\mathbf{W}^{\text{int}}$ and $\mathbf{W}^{\text{bdry}}$, introducing two trainable tensors $\mathbf{V}^{\text{int}}$ and $\mathbf{V}^{\text{bdry}}$

$$\mathbf{W}^{\text{int}}_{k_1\dots k_n i} = \frac{\exp \mathbf{V}^{\text{int}}_{k_1\dots k_n i}}{\sum_i \exp \mathbf{V}^{\text{int}}_{k_1\dots k_n i}},$$

$$\mathbf{W}^{\text{bdry}}_{k_1\dots k_n i} = \frac{\exp \mathbf{V}^{\text{bdry}}_{k_1\dots k_n i}}{\sum_i \exp \mathbf{V}^{\text{bdry}}_{k_1\dots k_n i}}.$$

For the sake of comparison, we examine three distinct methods for choosing convex combinations:

1. The individual entries of **V** are prescribed by directly trainable variables. In this case, the entries of **V** are initialized by sampling a standard normal distribution.

$$\mathbf{V}_{k_1\dots k_n i} = \theta_{k_1\dots k_n i} \tag{52}$$

2. The entries of **V** are prescribed by a ResNet with a Box initialization [19]. This network takes knot coordinates as input and outputs corresponding weights, i.e., $\mathbf{V}_{k_1\dots k_n i} = ResNet_i(t_{k_1},\dots,t_{k_n})$. We present results for a ResNet with four hidden layers, where each layer has a number of hidden nodes equal to twice the number of POUs, using *tanh* activation for the hidden layers and a linear final layer.

$$\begin{aligned} \mathbf{z}^{(0)}_{k_1\dots k_n} &= \sigma\left(\mathbf{A}^{(0)} t_{k_1\dots k_n} - \mathbf{b}^{(0)}\right) \\ \mathbf{z}^{(\ell)}_{k_1\dots k_n} &= \sigma\left(\mathbf{A}^{(\ell)} \mathbf{z}^{(\ell-1)}_{k_1\dots k_n} - \mathbf{b}^{(\ell)}\right) + \mathbf{z}^{(\ell)}_{k_1\dots k_n} \qquad \text{for layers } \ell = 1,\dots,L-1 \\ \mathbf{V}_{k_1\dots k_n i} &= \mathbf{A}^{(L)}_i \mathbf{z}^{(L)}_{k_1\dots k_n} - \mathbf{b}^{(L)}_i \end{aligned} \tag{53}$$

3. The entries of **V** are prescribed by the quadratic form associated with a trainable radial basis function (RBF) kernel. The parameterization learns centers $\mathbf{c}_i$ initialized via a uniform distribution over Ω and correlation factors $\mathbf{Q}_i$ initialized as identity matrices; here, the vector $t_{k_1,\dots,k_n}$ is the tensor product of individual spline knots $(t_{k_1},\dots,t_{k_n})$.

$$\mathbf{V}_{k_1\dots k_n i} = -(\mathbf{c}_i - t_{k_1\dots k_n})^T \mathbf{Q}^T \mathbf{Q} (\mathbf{c}_i - t_{k_1\dots k_n}) \tag{54}$$

Application of the softmax yields a RBF with POU normalization. Note that $\mathbf{Q}_i$ admit interpretation as Cholesky factors of a covariance matrix, allowing learning of anisotropic weights. If we instead took $\mathbf{Q}_i$ as a scalar q_i times an identity matrix, we would obtain a traditional RBF with q_i^2 prescribing a shape parameter [79].

While this choice of architectures is by no means complete, we hope to highlight a range of potential inductive biases to impose on the partitions. The first is most expressive, while the latter two impose more regularity and potentially are more memory efficient. The ResNet strategy may prove more effective for problems in high dimensions. We recommend readers to [70,46] for further discussion of POUNet architectures and their relative merits.

5.3.3. Constraints on exterior calculus metrics

The metric tensors $\mathbf{B}_0, \mathbf{B}_1, \mathbf{D}_0, \mathbf{D}_1$ must be symmetric-positive definite matrices to obtain a valid Hodge star operator and induce an inner-product [71]. For simplicity we parameterize these via diagonal matrices with positive entries. To obtain a parameterization of a metric tensor **M** (i.e., either $\mathbf{B}_k$ or $\mathbf{D}_k$), we introduce a trainable tensor **m**, and define

$$\mathbf{M} = \text{diag}(\exp(\mathbf{m})).$$

We denote by $\mathbf{b}_0, \mathbf{b}_1, \mathbf{d}_0, \mathbf{d}_1$ the parameters associated with $\mathbf{B}_0, \mathbf{B}_1, \mathbf{D}_0$ and $\mathbf{D}_1$, respectively. The trainable variables are all initialized as zero vectors to obtain an identity matrix initialization for the metric tensors, so d^0 and d^{0^T} initially match their combinatorial counterparts.

A more expressive alternative parameterization would be to learn the Cholesky factors $\mathbf{M} = \mathbf{m}\mathbf{m}^\intercal$, for upper-triangular trainable **m**, however we adopt a diagonal parameterization to promote sparsity.

5.3.4. Optimization problem statement

We denote the set of trainable variables ξ, which includes

$$\xi = \begin{cases} \text{distance between spline knot parameters } \{\tau^d_k \mid k = 1,\dots,N_{x_d}\} \\ \text{convex combination tensors } \mathbf{V}^{\text{int}}, \mathbf{V}^{\text{bdry}} \\ \text{metrics } \mathbf{b}_0, \mathbf{b}_1, \mathbf{d}_0, \mathbf{d}_1 \end{cases}. \tag{55}$$

With these trainable variables, Equation (38) specializes to

$$\begin{aligned} &\min_{\xi} \left\| p_{\text{data}} - \left(p_D + \sum_i \hat{p}_i \psi_i \right) \right\|_2^2 + \alpha^2 \left\| F_{\text{data}} - \sum_{ij} \hat{F}_{ij} \psi_{ij} \right\|_2^2 \\ &\text{such that } \begin{bmatrix} \mathbf{M}_1 & -\mathbf{M}_1 \mathbf{D}_1^{-1} \delta_0 \mathbf{D}_0 \\ -\mathbf{B}_0^{-1} \delta_0^T \mathbf{B}_1 \mathbf{M}_1 & \mathbf{0} \end{bmatrix} \begin{bmatrix} \widehat{\mathbf{F}} \\ \widehat{\mathbf{p}} \end{bmatrix} = \begin{bmatrix} \mathbf{b}_D \\ -\mathbf{b}_f - \mathbf{b}_N \end{bmatrix} \end{aligned} \tag{56}$$

where α is a normalization parameter used to ensure the flux and pressure loss are of the same magnitude; in our case, we choose

$$\alpha = \frac{\|p_{\text{data}}\|_2}{\|F_{\text{data}}\|_2}.$$

In Equation (55), the objective functions and constraints depend on ξ, in that the distance between spline knots τ_k^d and the convex combination tensors $\mathbf{V}^{\text{int}}$ and $\mathbf{V}^{\text{bdry}}$ parameterize the zero-forms ψ_i and one-forms ψ_{ij}, which are used as the bases in which our solution is expressed (they form the elements of the 0-form mass matrix $\mathbf{M}_1$ and the 1-form mass matrix $\mathbf{M}_1$), and the metrics $\mathbf{b}_0$, $\mathbf{b}_1$, $\mathbf{d}_0$, $\mathbf{d}_1$ parameterize the matrices $\mathbf{B}_0$, $\mathbf{B}_1$, $\mathbf{D}_0$, $\mathbf{D}_1$.

To solve this, we adopt a reduced-space strategy. The equality constraint amounts to a moderately sized dense linear system which may be inverted, providing

$$\widehat{\mathbf{p}} = -\mathbf{S}^{-1}\left(-\mathbf{B}_0^{-1}\delta_0^T\mathbf{B}_1\mathbf{b}_D + (\mathbf{b}_f + \mathbf{b}_N)\right) \tag{57}$$

$$\widehat{\mathbf{F}} = \mathbf{M}_1^{-1}\left(b_D + \mathbf{M}_1\mathbf{D}_1^{-1}\delta_0\mathbf{D}_0\widehat{\mathbf{p}}\right), \tag{58}$$

where $\mathbf{S} = \mathbf{B}_0^{-1}\delta_0^T\mathbf{B}_1\mathbf{M}_1\mathbf{D}_1^{-1}\delta_0\mathbf{D}_0$ is the Schur complement. These may be substituted into the objective of Equation (55) to obtain an unconstrained nonlinear optimization problem which may be treated with a first-order optimizer and back propagation. Alternatively for larger systems where inversion of $\mathbf{M}_1$ and $\mathbf{S}$ may be intractable, Lagrange multipliers may be applied as presented in [71].

Note that the loss function in this minimization problem is a data-sampled estimation of the squared $H^1(\Omega)$ norm; in our results, we refer to the first term of the objective as the $L^2(\Omega)$ loss and the unscaled second term as the $H_0^1(\Omega)$ seminorm loss.

6. Results

We consider the following problems:

1. Manufactured problem - establishing convergence properties
2. Five-strip problem - establishing structure preservation for $H(\text{div}, \Omega)$ problems
3. Battery problem - demonstrating surrogate construction for realistic multiscale problems

The manufactured problem serves as a comparison to standard methods for solving finite element problems; by manufacturing a solution we can quantify convergence with respect to fine and coarse resolution. The five-strip problem is a classical patch test for mixed FEM spaces to verify $H(\text{div}, \Omega$ conformity for problems with discontinuous material properties [35,72]; discretizations assuming regularity beyond $H(\text{div}, \Omega)$ exhibit oscillatory solutions. The battery problem models the flow of current through a lithium-ion matrix under a unit voltage drop. We solve both problems on a reference domain $\Omega = [0,1]^2$. Both problems obey the same set of differential equations (albeit with different coefficients, boundary conditions, forcing terms, etc.), presented below.

Data is taken either from randomly sampled evaluations of a known analytic solution, or if an analytic solution is not known (i.e., for the battery problem), high-fidelity FEM simulation (for more details, see Appendix B).

All training instances minimize Equation (56) using the Adam optimizer [39], with all parameters except the learning rate set according to the TensorFlow default values. The optimizer employed a learning rate schedule that reduced the learning rate by half if the loss did not improve after five epochs, terminating training after four such reductions of the learning rate. For the manufactured problem and five-strip problem, the batch size equalled the number of data points (i.e., each epoch consisted of one gradient descent step), while for the battery problem, the batch size was set to 100K points (i.e., each epoch consisted of 59 gradient descent steps to iterate through all 5.89M training points). Training was replicated with different random initializations five times (i.e., $N_{\text{samples}} = 5$) for each model instance. Code was run using TensorFlow 2.2 using a workstation running Linux RHEL 7.7 (Maipo) with 64 core Intel Xeon Gold 6130. Each training run used a single Nvidia Tesla V100 GPU with 32 GB memory. Unless otherwise noted, hyperparameters have been selected via a sequential grid search; the appendices include selected studies to demonstrate sensitivities to hyperparameters.

6.1. Manufactured problem

We first consider a manufactured problem with smooth solution. We consider the problem in Equation (43) on $\Omega = [0,1]^2$ under the forcing

$$\begin{aligned}
\mathbf{F} - \nabla p &= 0 && \text{in } \Omega \\
\nabla \cdot \mathbf{F} &= -8\pi^2 \sin(2\pi x)\sin(2\pi y) && \text{in } \Omega \\
p &= 0 && \text{on } \Gamma_D = \{(x,y) \in \Omega \mid x \in \{0,1\}\} \\
\mathbf{F} \cdot \vec{n} &= 0 && \text{on } \Gamma_N = \{(x,y) \in \Omega \mid y \in \{0,1\}\},
\end{aligned}$$

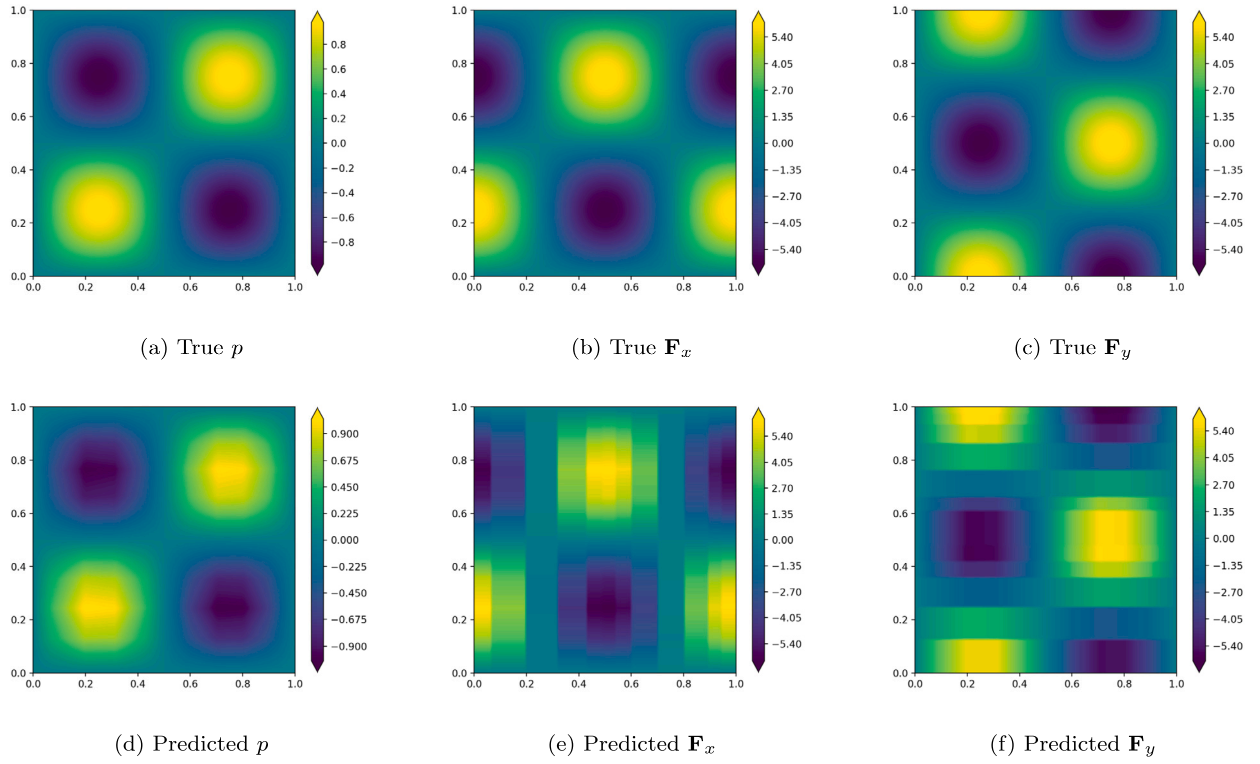


Fig. 3. Representative solutions for manufactured problem with 12×12 fine-scale resolution and 8 coarsened partitions. (Top) True solutions, for pressure p and flux $\mathbf{F}$; (Bottom) FEEC predictions.

Table 1
Error comparison between data-driven model and FEM for manufactured problem. FEEC method uses 4 interior POUs and 4 boundary POUs, resulting in a substantially smaller linear system while preserving comparable accuracy.

Method	Fine-scale resolution	Degrees of Freedom	Pressure MSE
FEM Q1	5×6	30	4.3470×10^{-3}
FEM Q1	4×8	32	7.0664×10^{-3}
FEM Q1	12×12	144	1.7891×10^{-4}
FEEC	12×12	32	2.1853×10^{-4}

which has the solution

$$\begin{aligned} p(x, y) &= \sin(2\pi x)\sin(2\pi y) \\ \mathbf{F}(x, y) &= \begin{bmatrix} 2\pi \cos(2\pi x)\sin(2\pi y) \\ 2\pi \sin(2\pi x)\cos(2\pi y) \end{bmatrix}. \end{aligned} \tag{59}$$

We solve (56) fitting to 10K points sampled from a uniform distribution over Ω. The loss function uses a weighting of $\alpha = \frac{1}{2\pi}$. We vary both the number of fine-scale resolutions and number of coarsened partitions to study convergence. We use a learning rate of 0.01 with the Adam optimizer, reducing the learning rate by half if the loss stagnates. Otherwise, the optimizer uses its default values in TensorFlow, training for a total of 1000 epochs. Unless otherwise noted, we consider the direct parameterization of the convex combination tensor.

6.1.1. *Model performance*

On this smooth problem, our FEEC method matches the performance of a nodal FEM at the fine scale resolution while requiring substantially fewer degrees of freedom. Representative results for training with 4 interior POUs and 4 boundary POUs are shown in Fig. 3. We compare the results of our trained FEEC model to three different FEM models using linear Lagrange elements on a quadrilateral mesh - one that maintains the same resolution as the POU's fine-scale grid, and two that match the number of degrees of freedom of the coarsened linear system. The FEM comparisons are computed using the FEniCSX software library [64,63,4]. Results are shown in Table 1. We achieve an accuracy that is comparable to solving at the same resolution of the fine-scale splines, while

Table 2
Error under refinement, increasing the number of fine-scale splines for the smooth sine-cosine problem. Convergence matches optimal convergence for piecewise linear continuous FEM.

Fine Scale	$L^2(\Omega)$ MSE			$H_0^1(\Omega)$ MSE		
Spline Grid	Median	Min	Max	Median	Min	Max
8×8	4.2083×10^{-4}	3.2706×10^{-4}	5.2920×10^{-4}	0.5559	0.5004	0.6171
12×12	1.4249×10^{-4}	1.1942×10^{-4}	1.5941×10^{-4}	0.3304	0.3049	0.3522
16×16	9.5777×10^{-5}	7.1075×10^{-5}	1.0280×10^{-4}	0.2672	0.2326	0.2790
20×20	6.8622×10^{-5}	4.1444×10^{-5}	8.6188×10^{-5}	0.2127	0.1672	0.2597
24×24	4.0407×10^{-5}	3.0599×10^{-5}	4.3923×10^{-5}	0.1595	0.1448	0.1775
28×28	2.6854×10^{-5}	1.9053×10^{-5}	4.4021×10^{-5}	0.1243	0.1053	0.1666
32×32	2.1173×10^{-5}	1.8414×10^{-5}	6.2870×10^{-5}	0.1025	0.0940	0.1552
Convergence rate	2.09	2.12	1.66	1.18	1.21	0.99

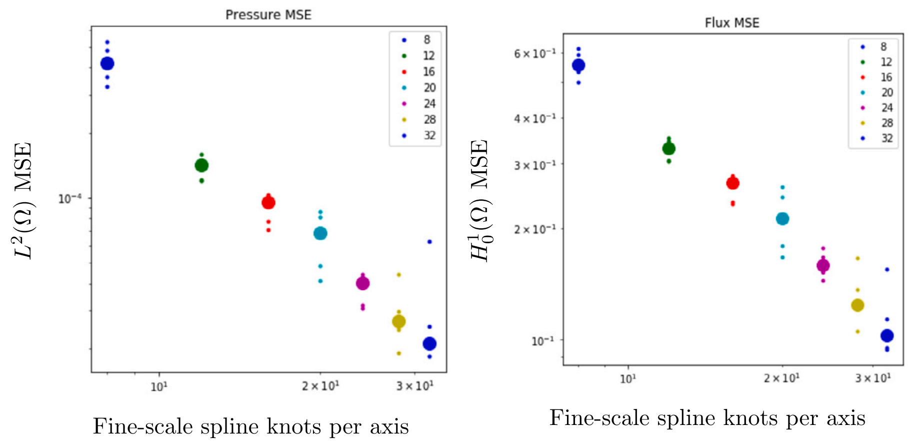


Fig. 4. Log-log plots for $L^2(\Omega)$ (left) and $H_0^1(\Omega)$ (right) loss terms, with an increasing number of fine-scale spline knots. For both terms of the loss, we observe a linear trend in our loss, matching the error rates of classical structure-preserving FEM methods.

Table 3
Training metrics when using different parameterizations of the convex combination tensor **W**.

Parameterization	$L^2(\Omega)$	$H_0^1(\Omega)$	$H^1(\Omega)$
Direct	$\mathbf{1.8088\times 10^{-4}}$	0.3663	0.3804
ResNet	2.3590×10^{-3}	0.3962	0.5801
RBFNet	4.8052×10^{-4}	**0.3299**	**0.3674**

solving a linear system substantially smaller than we would otherwise; similarly, we outperform the linear systems with comparable numbers of degrees of freedom by close to an order of magnitude, even for this small problem.

6.1.2. *Refinement study*

We next demonstrate convergence under h-refinement when increasing the number of fine-scale splines. Table 2 and Fig. 4 demonstrates second-order convergence under the $L^2(\Omega)$ norm and first-order convergence under the $H_0^1(\Omega)$ norm when refining the underlying B-spline grid, matching the optimal convergence of continuous Galerkin methods with linear polynomial basis. Each model is trained on the same dataset, using a fixed number of 8 internal POUs and 8 boundary POUs.

6.1.3. *Effect of POU parameterization*

We test the three different parameterizations of the convex combination tensor **W** described in Section 5.3.2, for a problem with a 12×12 fine-scale spline grid mapping to 8 interior POUs and 8 boundary POUs. Results are shown in Table 3. We note that training with the ResNet requires significantly longer to train (10K vs 1K epochs) at a smaller learning rate (10^{-4} vs. 10^{-2}). There was no noticeable difference in training time per epoch between each parameterization. Examples of representative POUs from each are shown in Fig. 5. While most expressive, training directly on the entries of **W** yields partitions with little regularity. ResNet's impose regularity but obtain globally supported partitions, while RBFs impose regularity and compactly supported partitions.

The question of when to anticipate compactly supported partitions is a subtle one. While a traditional FEM construction would generate compactly supported functions over elements, this is partly to ensure h-convergence under refinement. As our intention

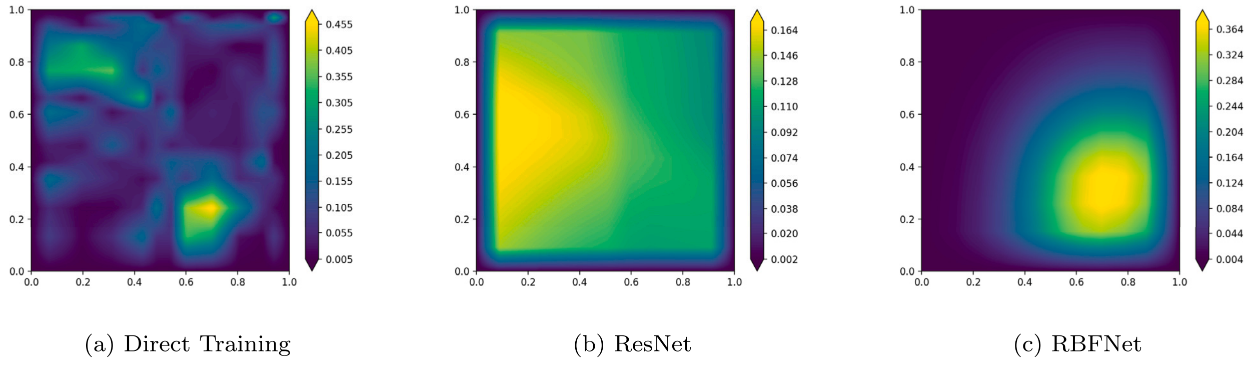


Fig. 5. A representative POU from each of the different parameterizations of the convex combination tensor **W**. Direct training is most expressive but imposes no regularity. ResNets and RBFNets impose regularity, with the RBFs biasing toward compactly supported partitions.

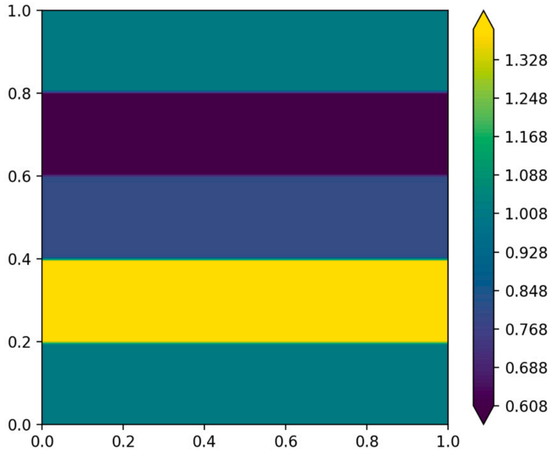


Fig. 6. Setup for diffusion coefficients for five strip problem. The gradient of the solution is perpendicular to the interface between strips, leading to a trivial piecewise linear solution.

is to use data to realize an accurate model on a *coarse* grid, this is not necessary. Intuition is further complicated by the fact that ResNets have no mechanism to minimize linear dependence between partitions; performing e.g., Gram-Schmidt to realize a maximally orthogonal basis would reveal a more compact representation. Regardless, the choice of parameterization has no effect on conservation.

6.2. *Five strip problem*

The five strip problem is a classical benchmark to test mixed methods for conservation laws with non-uniform diffusion coefficients: in each of five horizontal strips across $\Omega = [0,1]$, a different diffusion rate κ is assigned, as sketched in Fig. 6. Because the gradient of the solution p is perpendicular to the gradient in the material properties, the interface condition $[\![\kappa \partial_n p]\!] = 0$ is trivially satisfied.

The problem is given by

$$\begin{aligned}
\mathbf{F} - \boldsymbol{\kappa}\nabla p = 0 &\qquad \text{in } \Omega \\
\nabla \cdot \mathbf{F} = 0 &\qquad \text{in } \Omega \\
p = 0 &\qquad \text{on } \Gamma_{D,\text{left}} = \{(0,y)\} \\
p = 1 &\qquad \text{on } \Gamma_{D,\text{right}} = \{(1,y)\} \\
\mathbf{F} \cdot \vec{n} = 0 &\qquad \text{on } \Gamma_N = \{(x,y) \in \Omega \mid y \in \{0,1\}\},
\end{aligned}$$

where we define

$$\boldsymbol{\kappa}(x,y) = \begin{cases} 1.0 & x \in \left[0, \frac{1}{5}\right] \\ 1.4 & x \in \left[\frac{1}{5}, \frac{2}{5}\right] \\ 0.8 & x \in \left[\frac{2}{5}, \frac{3}{5}\right] \\ 0.6 & x \in \left[\frac{3}{5}, \frac{4}{5}\right] \\ 1.0 & x \in \left[\frac{4}{5}, 1\right] \end{cases}$$

This problem has the unique solution

$$\begin{aligned} p(x,y) &= x \\ \mathbf{F}(x,y) &= \begin{cases} \begin{bmatrix} 1.0 \\ 0 \end{bmatrix} & \text{for } x \in \left[0, \frac{1}{5}\right] \\ \begin{bmatrix} 1.4 \\ 0 \end{bmatrix} & \text{for } x \in \left[\frac{1}{5}, \frac{2}{5}\right] \\ \begin{bmatrix} 0.8 \\ 0 \end{bmatrix} & \text{for } x \in \left[\frac{2}{5}, \frac{3}{5}\right] \\ \begin{bmatrix} 0.6 \\ 0 \end{bmatrix} & \text{for } x \in \left[\frac{3}{5}, \frac{4}{5}\right] \\ \begin{bmatrix} 1.0 \\ 0 \end{bmatrix} & \text{for } x \in \left[\frac{4}{5}, 1\right] \end{cases} \end{aligned} \tag{60}$$

We train our FEEC model with a fine-scale resolution of 2×20 B-splines, coarsened into 5 interior POUs and 5 exterior POUs. As we have chosen a number of POUs equal to the number of strips, this test serves to identify whether we can exactly recover the five physically relevant subdomains as control volumes. We use an initial learning rate of $\lambda = 0.01$, which is reduced by half if the loss stagnates. We train for 5000 epochs on a dataset of 10K uniformly sampled points in $\Omega = [0,1]^2$, with a batch size equal to the dataset size.

The results in Fig. 7 demonstrate the ability of the scheme to reproduce the solution for both $\mathbf{F}$ and p, indicating that it has correctly identified each strip as a control volume. These results are particularly notable since the FEEC method is provided no information about the location of the strips, nor the values of the corresponding diffusion coefficients; in an unsupervised fashion the geometry is learned by the POU parameterization and the material property is learned by the metric information. We note that for this problem, the adaptivity of the spline knots is crucial: during training, the spline knots are pushed towards the interfaces of the strips, to provide the sharp interfaces that appear in Fig. 7.

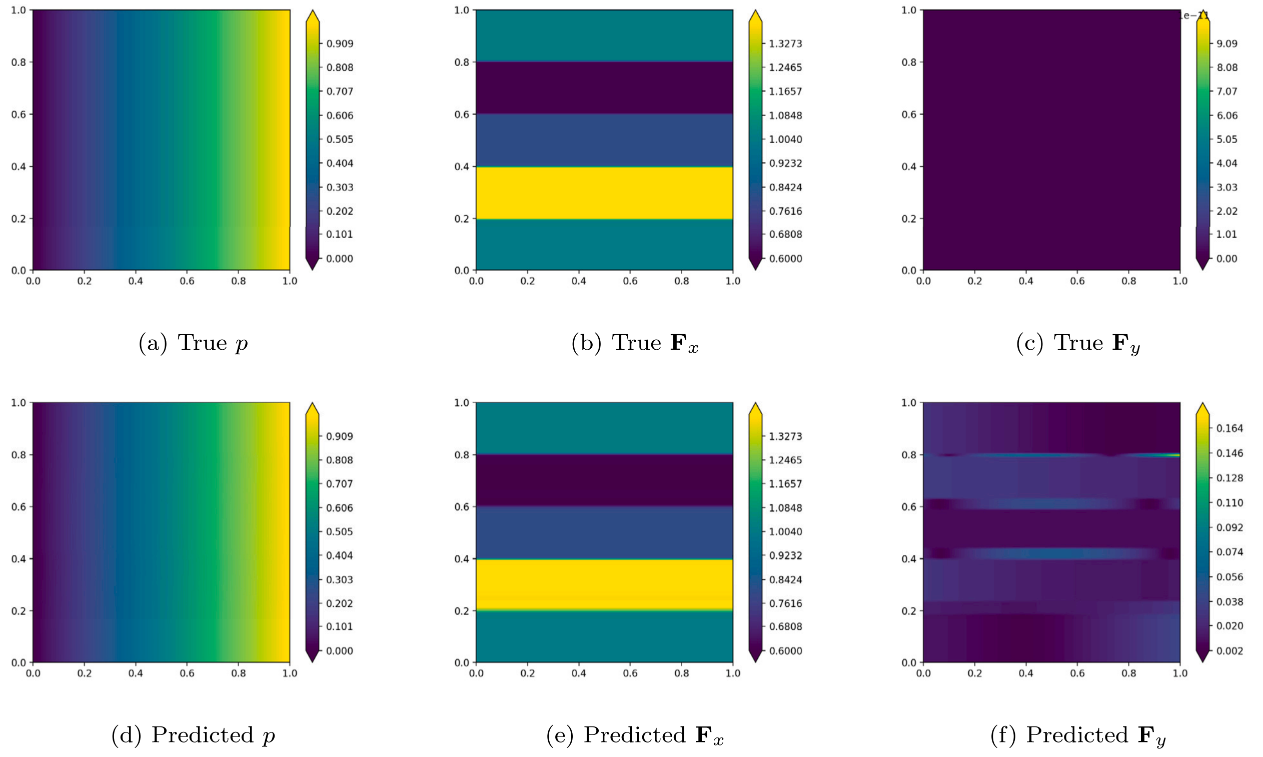
6.3. Battery problem

The battery problem provides a nontrivial extension of the five-strip problem where now the geometry is complex and the material coefficients vary over seven orders of magnitude across different subdomains, providing a pathologically ill-conditioned problem representative of multiscale problems. Data is provided by a combination of discrete element method and conformal finite element codes to generate a realistic microstructure corresponding to a lithium-ion battery [60,68]; details of the simulation are provided in Appendix B. We consider

$$\begin{aligned} \mathbf{F} - \kappa \nabla p &= 0 && \text{in } \Omega \\ \nabla \cdot \mathbf{F} &= 0 && \text{in } \Omega \\ p &= 1 && \text{on } \Gamma_{D,\text{left}} = \{(0,y)\} \\ p &= 0 && \text{on } \Gamma_{D,\text{right}} = \{(1,y)\} \\ \mathbf{F} \cdot \vec{n} &= 0 && \text{on } \Gamma_N = \{(x,y) \mid y \in \{0,1\}\}. \end{aligned}$$

The values for κ follow a configuration of three phases of materials - an active lithium phase ($\kappa = 1$), a conductive binder ($\kappa = 100$), and an electrolyte material ($\kappa = 0.0001$). The configuration for κ is shown in Fig. 8. Similar to the five-strip problem, successfully fitting control volumes to this problem corresponds to identifying the topologically complex transport pathways associated with the microstructure.

We train our FEEC model with a learning rate $\lambda = 0.05$, number of interior POUs $N^0_{\text{int}} = 8$, and exterior POUs $N^0_{\text{bdry}} = 8$. Results showing the sensitivities of hyperparameters are presented in Appendix C.


(a) True p (b) True $\mathbf{F}_x$ (c) True $\mathbf{F}_y$

(d) Predicted p (e) Predicted $\mathbf{F}_x$ (f) Predicted $\mathbf{F}_y$

Fig. 7. Results from data-driven model for the five strip problem. (Top) True solutions, for pressure p and flux $\mathbf{F}$; (Bottom) FEEC predictions. By recovering the true solution using the same number of partitions as strips, we demonstrate our ability to recover physically relevant subdomains in an unsupervised manner, without requiring data for strip geometry or material properties.

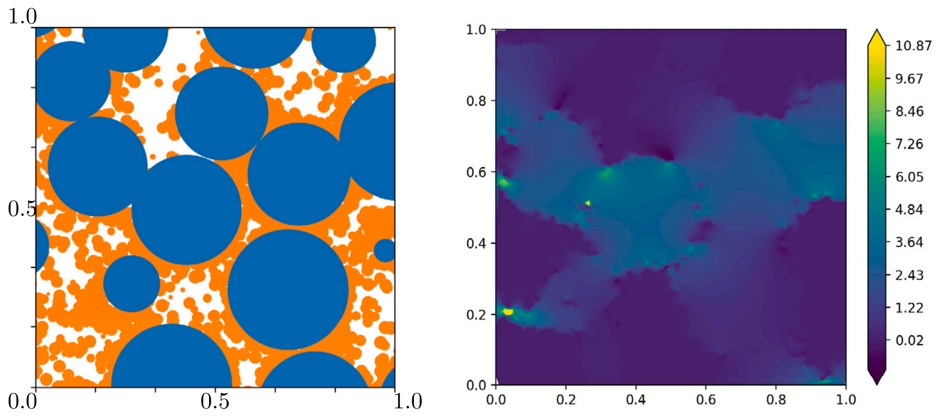


Fig. 8. Microstructure configuration for lithium-ion battery problem (left). Lithium particles (blue) are embedded in a non-conductive matrix (white), with a conductive binder (orange) promoting diffusion through microstructure. Transport is governed by *transport pathways* corresponding to a percolation problem, and provides a realistic assessment of whether we can identify associated control volumes. Associated electric field (right, x-component shown) demonstrates singularities occurring at junctions between phases, presenting challenging small scale features to recover. (For interpretation of the colors in the figure(s), the reader is referred to the web version of this article.)

6.3.1. Model performance

Representative solutions and comparisons to true solutions for p and $\mathbf{F}$ are shown in Fig. 9. Additionally, we can preserve quantities of interest for this problem, such as the flux out the right-hand boundary, within 1% error after hyperparameter turning, while reducing the size of the underlying linear system from 5.89M degrees of freedom to 136, corresponding to a coarsening from a 20×20 fine-scale spline knot grid to 8 interior POUs and 8 boundary POUs. Descriptive statistics for the most accurate training hyperparameters are reported in Table 4; in Appendix C we study how specific hyperparameters influence training accuracy.

For the battery problem, the flux through the domain boundary (i.e., integrating the normal component of $\mathbf{F}$ along the line $\{x = 1\}$) is an important quantity of interest associated with the effective diffusion coefficient of the microstructure. We achieve a

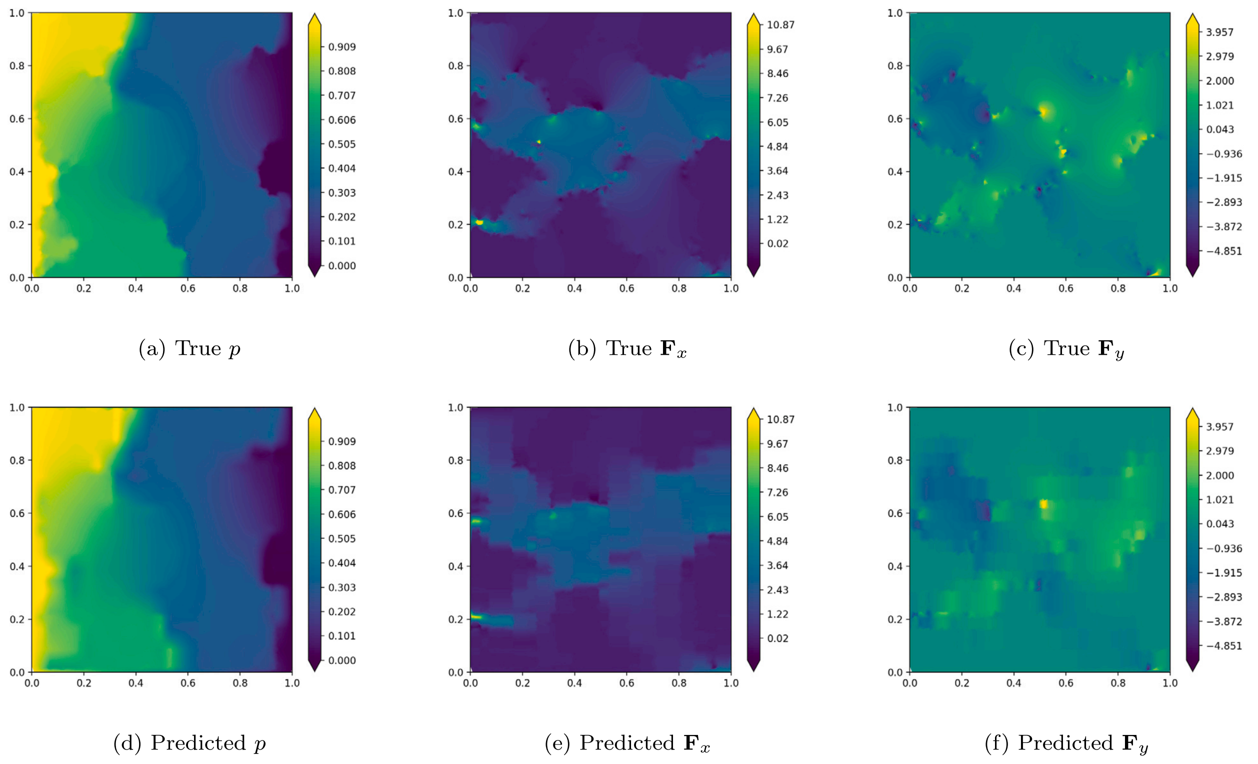


Fig. 9. True and predicted potential and electric fields for lithium-ion battery microstructure shown in Fig. 8. Predictions use a 20×20 B-spline grid, with 8 POUs for the interior and an additional 8 POUs for the exterior.

Table 4
Best resulting accuracy across all sample runs and hyperparameters for $L^2(\Omega)$, $H^1_0(\Omega)$, and $H^1(\Omega)$ errors (left) and for the quantity of interest associated with the battery problem (right). Training was performed for 20×20 fine-scale spline knots coarsened to 8 interior and 8 exterior POUs. Reduction of the data from 5.89 million degrees of freedom to 136 degrees of freedom amounts to a 43308$\times$ reduction in model size while preserving 1% error for relevant quantity of interest.

Metric	Best Error
$L^2(\Omega)$	5.206×10^{-4}
$H^1_0(\Omega)$	1.655×10^{-1}
$H^1(\Omega)$	1.708×10^{-1}

Flux Through Boundary	
Predicted Value	29.0910
True Value	29.3893
Relative Error	1.0147%

best relative error of 1%, with summary statistics presented in Table 4. Accurate prediction of this quantity is notable, as there are no training data used to directly recover it and it instead follows from correctly preserving conservation structure. This is in contrast to physics-informed approaches which incorporate this data directly and impose it by penalty (e.g., [36,81]). In Appendix C we include a study exploring how this quantity of interest and full field error are effected by hyperparameters such as the learning rate and the number of fine-scale splines. In Fig. 10 we study h-refinement of the B-spline grid and its effect on $L^2(\Omega)$ and $H^1_0(\Omega)$ error. While there is more variance during training due to random initialization, we generally approach a similar second- and first-order convergence rate trend for pressure and flux as seen in the manufactured solution study, as shown in Table 5.

6.4. Conclusions

We have presented a new unsupervised learning architecture for discovering structure-preserving control volume models from data. A differentiable construction of Whitney forms with respect to an underlying machine learned partition of unity parameterizes a de Rham complex, inducing learnable discrete coboundary operators which provide a link between subdomain geometry and graphs encoding flux/circulation balances. The primary contribution is an end-to-end differentiable parameterization of geometry and physics on arbitrary manifolds. By applying equality constrained optimization, we simultaneously obtain partitions of space and accompanying integral balance laws which match data. Current methods for physics-informed learning rely on collocation or

Table 5
Resulting accuracy for pressure, i.e., $L^2(\Omega)$, error (left) and flux, i.e., $H_0^1(\Omega)$, error (right). Training was performed for 20×20 fine-scale spline knots coarsened to 8 interior and 8 exterior POUs with a learning rate of $\lambda = 0.05$. Reduction of the data from 5.89 million degrees of freedom to 136 degrees of freedom amounts to a 43308× reduction in model size while still maintaining expected convergence rates.

Fine Scale	$L^2(\Omega)$ MSE			$H_0^1(\Omega)$ MSE		
Spline Grid	Median	Min	Max	Median	Min	Max
8×8	4.782×10^{-3}	3.888×10^{-3}	5.334×10^{-3}	0.3626	0.3038	0.3906
12×12	3.846×10^{-3}	2.414×10^{-3}	5.189×10^{-3}	0.2688	0.2651	0.3196
16×16	2.601×10^{-3}	1.417×10^{-3}	4.702×10^{-3}	0.2309	0.2214	0.2584
20×20	1.786×10^{-3}	1.428×10^{-3}	3.453×10^{-3}	0.1984	0.1832	0.2068
24×24	1.192×10^{-3}	1.030×10^{-3}	2.357×10^{-3}	0.1700	0.1495	0.1832
Convergence rate	1.25	1.19	0.69	0.67	0.64	0.71

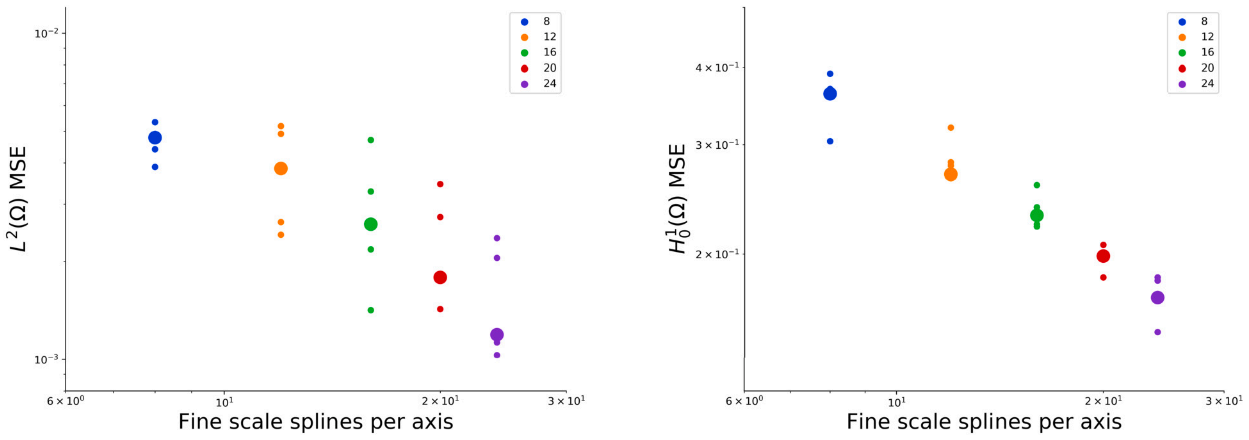


Fig. 10. $L^2(\Omega)$ (left) and $H_0^1(\Omega)$ (right) loss terms for an increasing number of fine-scale spline knots. For both terms of the loss, we observe a linear trend in our loss, matching the error rates of classical structure-preserving FEM methods.

other discretizations of partial differential operators to fit data via backpropagation, whereas the geometric approach presented here naturally preserves related topological structure.

The approach offers several benefits which we pursue in future work, both as a framework for learning computationally efficient reduced-order models and as a tool for learning data-driven models when first-principles derivation is intractable. Combining this work with the nonlinear extensions in [71] allows the construction of reduced-order models without reference to an underlying full-order model; this avoids the need for hyper-reduction and remains applicable when the full governing equations are unknown. Additionally, the learned model provides a natural physics-preserving surrogate of Dirichlet2Neumann maps, which have proven to be a natural means of describing multiscale/domain decomposition techniques such as FETI [23] and mortar methods [13,5]. The technique is therefore an ideal candidate for building surrogates in multiscale systems-of-systems, replacing component models in systems with either partially unknown or computationally intractable physics.

For traditional forward simulation, the de Rham complex is primarily a tool for studying second-order linear elliptic equations with limited applicability beyond an important but limited class of problems, including: Maxwell, transport physics, and mechanics. The data-driven setting potentially extends much broader, as we have shown in [71] that a data-driven Hodge Laplacian may be used to stabilize a "black-box" unknown flux, ensuring structure-preservation and well-posedness while take advantage of deep networks ability to capture arbitrary flux relationships. An important and interesting aspect of future work will focus on how to include other aspects of structure-preservation for advection-dominated problems which require notions of compatibility related to bounds preservation and Lie derivatives not provided by the de Rham theory.

A particular promising direction of research is whether alternative parameterizations may be developed beyond the coarsened B-splines pursued here. Of particular interest are parameterizations which scale to large dimensions while maintaining approximation properties and tractable quadrature rules. In principle, POUs could be parameterized by e.g., a multilayer perceptron composed with a softmax activation; however this forces the use of Monte Carlo quadrature and subsequently incurs a variational crime. While we have avoided this in favor of a more rigorous mathematical setting, in engineering contexts these and other alternative parameterizations may have substantial value. This is particularly true for high-dimensional problems where the current B-spline construction will suffer from the curse-of-dimensionality (i.e., computational complexity scaling as the number of 1D splines raised to the dimension).

While we have presented the Whitney form construction for a general manifold, the current work applies the theory only to compact subsets of $\mathbb{R}^2$. There are several promising directions where the general theory is likely to prove impactful. For some applications, models may be learned directly on manifolds: e.g., atmosphere physics on the sphere, transport dynamics on biomembranes, or shell models for elastoplasticity could all be developed rigorously from this theory. For high-dimensional PDEs, one could pursue dimension reduction by developing autoencoder architectures to identify models on smooth manifolds. We pursue these lines of research to future work.

CRediT authorship contribution statement

Jonas A. Actor: Methodology, Software, Validation, Visualization, Writing. **Xiaozhe Hu:** Conceptualization, Methodology, Formal Analysis, Writing. **Andy Huang:** Conceptualization, Methodology, Writing. **Scott Roberts:** Validation, Data curation, Writing. **Nathaniel Trask:** Supervision, Conceptualization, Methodology, Writing, Funding acquisition.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

Data will be made available on request.

Acknowledgements

Sandia National Laboratories is a multimission laboratory managed and operated by National Technology & Engineering Solutions of Sandia, LLC, a wholly owned subsidiary of Honeywell International Inc., for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-NA0003525. This paper describes objective technical results and analysis. Any subjective views or opinions that might be expressed in the paper do not necessarily represent the views of the U.S. Department of Energy or the United States Government. This article has been co-authored by an employee of National Technology & Engineering Solutions of Sandia, LLC under Contract No. DE-NA0003525 with the U.S. Department of Energy (DOE). The employee owns all right, title and interest in and to the article and is solely responsible for its contents. The United States Government retains and the publisher, by accepting the article for publication, acknowledges that the United States Government retains a non-exclusive, paid-up, irrevocable, world-wide license to publish or reproduce the published form of this article or allow others to do so, for United States Government purposes. The DOE will provide public access to these results of federally sponsored research in accordance with the DOE Public Access Plan https://www.energy.gov/downloads/doe-public-access-plan.

N. Trask and J. A. Actor's work is supported by the U.S. Department of Energy, Office of Advanced Scientific Computing Research under the "Scalable and Efficient Algorithms - Causal Reasoning, Operators and Graphs" (SEA-CROGS) project and the Early Career Research Program. A. Huang is supported under the Sandia National Laboratories Laboratory Directed Research and Development (LDRD) program. The work of X. Hu is partially supported by the National Science Foundation under grant DMS-2208267.

Appendix A. Exact integration

We show how to assemble the matrices $\mathbf{M}_1$ and DIV^T using exact expressions for the integrals involved in their definitions. We emphasize that we have closed-form expressions for all of our integrals, without resorting to quadrature, but leave the remainder as an exercise for the reader. For simplicity, we assume that $\Omega = [0,1]^2$.

A.1. Notation

In the description that follows, we use the following indexing conventions:

- the letter c indexes fine-scale cells, i.e., sub-intervals of $[0,1]$ between two fine-scale spline knots
- the letters k, l, m, and n index 1D splines and spline knots
- the multi-index $\boldsymbol{k}$ indexes a 2D tensor product of 1D spline functions
- the single letters i and j index POU functions, i.e., 0-forms
- the pairs of letters ij index 1-forms
- the single letter a indexes 0-form test functions
- the pair of letters ab index 1-form test functions

When clear, summation bounds are omitted.

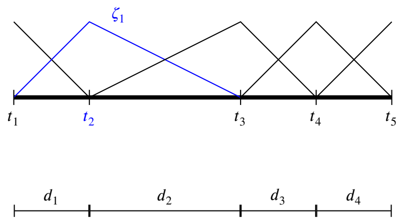


Fig. A.11. Example of one-dimensional B-splines ζ_k for $k = 1, \dots, N = 5$. Knots t_k and distances d_k are correspondingly labeled.

A.2. 1D assembly

We begin by introducing closed-form expressions for the integral of products of B-spline functions in one dimension, as well as their derivatives, to arbitrary powers. We denote the one-dimensional B-spline functions as ζ_k for $k = 1, \dots, N$; for example, B1 spline basis functions (i.e., hat functions) $\zeta_k : [0,1] \to [0,1]$ are defined as

$$\zeta_k(x) = \begin{cases} 0 & x < t_{k-1} \\ \frac{x - t_{k-1}}{t_k - t_{k-1}} & x \in [t_{k-1}, t_k] \\ \frac{t_k - x}{t_{k+1} - t_k} & x \in [t_k, t_{k+1}] \\ 0 & x > t_{k+1} \end{cases},$$

where $\{t_1 = 0, t_2, \dots, t_{N-1}, t_N = 1\}$ are spline knots. Hat functions are indexed so that they provide a basis to interpolate the spline knots, i.e., $\zeta_k(t_l) = \delta_{kl}$ (Fig. A.11). On each subinterval $d_c = [t_{c-1}, t_c]$ we define

$$\begin{aligned} I_c(\alpha, \beta, \eta, \nu) := &\int_{t_{c-1}}^{t_c} \left(\partial\zeta_{c-1}\right)^\alpha \left(\partial\zeta_c\right)^\beta \zeta_{c-1}^\eta \zeta_c^\nu \\ &= (-1)^\alpha \frac{1}{\eta + \nu + 1} \frac{1}{\binom{\eta+\nu}{\eta}} d_c^{1-(\alpha+\beta)}, \end{aligned} \tag{A.1}$$

where $\binom{\eta+\nu}{\eta}$ denotes the expression for the binomial coefficient. Thus,

$$\int_0^1 \left(\partial\zeta_{c-1}\right)^\alpha \left(\partial\zeta_c\right)^\beta \zeta_{c-1}^\eta \zeta_c^\nu = \sum_c I_c(\alpha, \beta, \eta, \nu). \tag{A.2}$$

With this expression, we build four-tensors that we will later use to assemble the matrices $\mathbf{M}_1$ and DIV^T. Along each dimension $z \in \{x, y\}$, we define four-tensors $\mathbf{A}^z$ and $\mathbf{C}^z$ of size $N \times N \times N \times N$, defined entry-wise via

$$\mathbf{A}^z_{klmn} = \int_0^1 \zeta^z_k(z)\ \zeta^z_l(z)\ \zeta^z_m(z)\ \zeta^z_n(z)\ dz$$

$$\mathbf{C}^z_{klmn} = \int_0^1 \zeta^z_k(z)\ \partial_z\zeta^z_l(z)\ \zeta^z_m(z)\ \partial_z\zeta^z_n(z)\ dz.$$

We may express both integrals as sums of $I_c(\alpha, \beta, \eta, \nu)$. First, for $\mathbf{A}^z_{klmn}$, we observe that each entry can be rewritten as

$$\mathbf{A}^z_{klmn} = \sum_c I_c(0, 0, \eta_c, \nu_c) \mathbf{1}_{\eta_c + \nu_c = 4}, \tag{A.3}$$

where

$$\begin{aligned} \eta_c &= \#\{j = c - 1 \mid j \in \{k, l, m, n\}\} \\ \nu_c &= \#\{j = c \qquad \mid j \in \{k, l, m, n\}\}. \end{aligned} \tag{A.4}$$

When we evaluate the $I_c(0, 0, \eta_c, \nu_c)$ terms, we arrive at the following values:

$$\mathbf{A}^z_{klmn} = \begin{cases} \frac{1}{5} d^z_1 & k=l=m=n=0 \\ \frac{1}{5}(d^z_k + d^z_{k+1}) & k=l=m=n \in \{1,\dots,N-1\} \\ \frac{1}{5} d^z_N & k=l=m=n=N \\ \frac{1}{20} d^z_{k+1} & \begin{cases} k=l=m,\ n=k+1,\ k\in\{0,\dots,N-1\} \\ k=l=n,\ m=k+1,\ k\in\{0,\dots,N-1\} \\ k=m=n,\ l=k+1,\ k\in\{0,\dots,N-1\} \\ l=m=n,\ k=l+1,\ k\in\{0,\dots,N-1\} \\ l=m=n=k+1,\ k\in\{0,\dots,N-1\} \\ k=m=n=l+1,\ k\in\{0,\dots,N-1\} \\ k=l=n=m+1,\ k\in\{0,\dots,N-1\} \\ k=l=m=n+1,\ k\in\{0,\dots,N-1\} \end{cases} \\ \frac{1}{30} d^z_{k+1} & \begin{cases} k=l,\ m=n=k+1,\ k\in\{0,\dots,N-1\} \\ k=m,\ l=n=k+1,\ k\in\{0,\dots,N-1\} \\ k=n,\ l=m=k+1,\ k\in\{0,\dots,N-1\} \\ l=m,\ k=n=l+1,\ k\in\{0,\dots,N-1\} \\ l=n,\ k=m=l+1,\ k\in\{0,\dots,N-1\} \\ m=n,\ k=l=m+1,\ k\in\{0,\dots,N-1\} \end{cases} \end{cases}.$$

Similarly,

$$\mathbf{C}^z_{klmn} = \sum_c I_c(\alpha_c, \beta_c, \eta_c, \nu_c) \mathbf{1}_{\eta_c+\nu_c=2, \alpha_c+\beta_c=2} \tag{A.5}$$

where

$$\begin{aligned} \alpha_c &= \#\{j = c-1 \mid j \in \{l,n\}\} \\ \beta_c &= \#\{j = c \quad\ \mid j \in \{l,n\}\} \\ \eta_c &= \#\{j = c-1 \mid j \in \{k,m\}\} \\ \nu_c &= \#\{j = c \quad\ \mid j \in \{k,m\}\}. \end{aligned} \tag{A.6}$$

Computing these values, we find

$$\mathbf{C}^z_{klmn} = \begin{cases} \frac{1}{3}\frac{1}{d^z_1} & k=l=m=n=1 \\ \frac{1}{3}\left(\frac{1}{d^z_k} + \frac{1}{d^z_{k+1}}\right) & k=l=m=n \in \{2,\dots,N-1\} \\ \frac{1}{3}\frac{1}{d^z_N} & k=l=m=n=N \\ \frac{-1}{3}\frac{1}{d^z_k} & \begin{cases} k=l=m,\ n=k+1,\ k\in\{1,\dots,N-1\} \\ k=m=n,\ l=k+1,\ k\in\{1,\dots,N-1\} \\ k=m=n=l+1,\ k\in\{1,\dots,N-1\} \\ k=l=m=n+1,\ k\in\{1,\dots,N-1\} \end{cases} \\ \frac{1}{6}\frac{1}{d^z_k} & \begin{cases} k=l=n,\ m=k+1,\ k\in\{1,\dots,N-1\} \\ l=m=n,\ k=l+1,\ k\in\{1,\dots,N-1\} \\ l=m=n=k+1,\ k\in\{1,\dots,N-1\} \\ k=l=n=m+1,\ k\in\{1,\dots,N-1\} \end{cases} \\ \frac{-1}{6}\frac{1}{d^z_k} & \begin{cases} k=l,\ m=n=k+1,\ k\in\{1,\dots,N-1\} \\ k=n,\ l=m=k+1,\ k\in\{1,\dots,N-1\} \\ l=m,\ k=n=l+1,\ k\in\{1,\dots,N-1\} \\ m=n,\ k=l=m+1,\ k\in\{1,\dots,N-1\} \end{cases} \\ \frac{1}{3}\frac{1}{d^z_k} & \begin{cases} k=m,\ l=n=k+1,\ k\in\{1,\dots,N-1\} \\ l=n,\ k=m=l+1,\ k\in\{1,\dots,N-1\} \end{cases} \end{cases}.$$

Note that both of these tensors $\mathbf{A}^z$ and $\mathbf{C}^z$ are both sparse and structured.

A.3. 2D assembly

We next to assemble the mass-matrix of one-forms, $\mathbf{M}_1$. Consider the entry $\mathbf{M}_{1(ab),(ij)}$, which is given as

$$\begin{aligned}\mathbf{M}_{1(ab),(ij)} &= \int_\Omega \psi_{ij} \cdot \psi_{ab} \\ &= \int_\Omega (\psi_i \partial_x \psi_j - \psi_j \partial_x \psi_i)(\psi_a \partial_x \psi_b - \psi_b \partial_x \psi_i) \\ &\qquad + (\psi_i \partial_y \psi_j - \psi_j \partial_y \psi_i)(\psi_a \partial_y \psi_b - \psi_b \partial_y \psi_i).\end{aligned}$$

This integral decouples into the product of two integrals, one over the domain $x = [0, 1]$ and one over $y = [0, 1]$. For example, the first cross-term becomes:

$$\begin{aligned}&\int_\Omega \psi_i \left(\partial_x \psi_j\right) \psi_a \left(\partial_x \psi_b\right) \\ &= \int_\Omega \left(\sum_{\boldsymbol{k}_i=(k_i,l_i)} \mathbf{W}_{\boldsymbol{k}_i i}\, \zeta_{\boldsymbol{k}_i}(\boldsymbol{x}) \right) \cdot \partial_x \left(\sum_{\boldsymbol{k}_j=(k_j,l_j)} \mathbf{W}_{\boldsymbol{k}_j j}\, \zeta_{\boldsymbol{k}_j}(\boldsymbol{x}) \right) \cdot \\ &\qquad \left(\sum_{\boldsymbol{k}_a=(k_a,l_a)} \mathbf{W}_{\boldsymbol{k}_a a}\, \zeta_{\boldsymbol{k}_a}(\boldsymbol{x}) \right) \cdot \partial_x \left(\sum_{\boldsymbol{k}_b=(k_b,l_b)} \mathbf{W}_{\boldsymbol{k}_b b}\, \zeta_{\boldsymbol{k}_b}(\boldsymbol{x}) \right) d\boldsymbol{x} \\ &= \sum_{\boldsymbol{k}_i} \sum_{\boldsymbol{k}_j} \sum_{\boldsymbol{k}_a} \sum_{\boldsymbol{k}_b} \mathbf{W}_{\boldsymbol{k}_i i} \mathbf{W}_{\boldsymbol{k}_j j} \mathbf{W}_{\boldsymbol{k}_a a} \mathbf{W}_{\boldsymbol{k}_b b} \cdot \mathbf{C}^x_{k_i k_j k_a k_b} \cdot \mathbf{A}^y_{l_i l_j l_a l_b}.\end{aligned} \tag{A.7}$$

We can build similar expressions for the other seven terms in the integrand of the expression for $\mathbf{M}_{1(ab),(ij)}$, expressing the integrand as tensor contractions involving permutations of $\mathbf{W}$, $\mathbf{C}^z$, and $\mathbf{A}^z$.

The matrix DIV and the vectors on the right-hand side of the linear system in Equation (46) can also be assembled using a similar process of decoupling the integral into the product of two one-dimensional integrals and then combining the one-dimensional integrals using tensor contraction. Alternatively, since $\mathrm{DIV}^T = \mathbf{M}_1 \delta_0$, once $\mathbf{M}_1$ is assembled it may be cheaper to assemble the combinatorial gradient δ_0 and construct DIV via matrix products instead. Note that the evaluation of the right-hand side vectors in Equation (46) require the ability to evaluate the functions f, g_D and g_N at the fine-scale spline knots, which evolve during the course of training; these functions are approximated by their fine-scale piecewise linear interpolants at the spline knots.

We make a handful of observations about the assembly process. First, we immediately note that since the tensor $\mathbf{W}$ and the distance between spline knots d^z evolve with each gradient descent step, assembly must be repeated at every step during training. The second observation concerns the memory cost of performing tensor contraction for entries in $\mathbf{M}_{1(ab),(ij)}$. While the tensors $\mathbf{A}^z$ and $\mathbf{C}^z$ are sparse, the TensorFlow operators for tensor contraction only accept dense tensors. As a result, the tensor contraction in e.g., Equation (A.7) requires an intermediate step that allocates a dense tensor of size $N_x^2 N_y^2 {N^0}^2$, as an in-place memory accumulator, which becomes limiting when the number of splines at fine-scale resolution along each axis becomes large (i.e., values of $N \geq 40$). Alternatively, the tensor contraction can be expressed by the appropriate sparse-dense matrix multiplication operations, although this method ultimately does not save any memory usage - the 1-form fine-to-coarse map is of size $N_x^2 N_y^2 {N^0}^2$, which is of the same size as the intermediate allocation step.

Appendix B. Numerical details for battery problem

The lithium ion battery cathode configuration depicted in Fig. 8 is composed of three phases, active material particles shown in blue, a conductive binder domain shown in orange, and an electrolyte shown in white. These configurations were generated using the discrete element method implemented in LAMMPS [69], the details of which are found in [68]. This discrete element method data was used to generate a physically realistic microstructure, and was then remapped onto an unstructured tetrahedral mesh using the conformal decomposition finite element method. For further details we refer readers to [60,68].

For this problem, the data in the loss comes from a high-fidelity continuous Galerkin finite element method implemented in Sierra/Aria [66]. An electrical transport equation

$$\nabla \cdot (\kappa \nabla V) = 0 \tag{B.1}$$

is solved on the domain for voltage V on the nodal locations, where the active material particles are assigned $\kappa = 1$, the conductive binder domain particles $\kappa = 100$, and the electrolyte $\kappa = 0.0001$. Piecewise linear integration functions on quadrilaterals were used. Dirichlet boundary conditions of $V = 1$ and $V = 0$ are assigned to the left and right hand sides of the domain, respectively. The equation is integrated by parts, assembled into a linear system, and solved iteratively in parallel using a GMRES solver with multi-level preconditioner, all implemented through Trilinos [73]. Fluxes are reconstructed as piecewise constant on the elements (P0) directly from the calculated voltage fields.

Table C.6
List of model parameters and training hyperparameters that were varied for battery problem.

	Parameter	Values
N_x, N_y	Fine-scale spline resolution	$\{8, 12, 16, 20, 24\}$
N^0_{int}	Coarse-scale interior POUs	$\{4, 8, 12, 16\}$
λ	Learning Rate	$\{0.001, 0.005, 0.05\}$

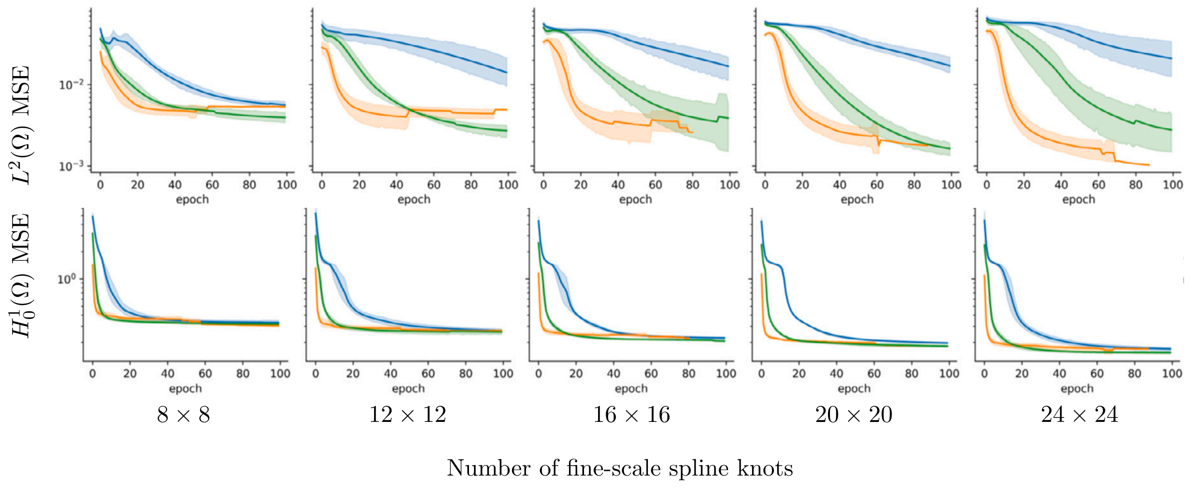


Fig. C.12. Loss curves for $L^2(\Omega)$ (top) and $H^1_0(\Omega)$ (bottom) components of the overall loss function, plotted after each epoch of training, for an increasing number of fine-scale spline nodes. Each color represents a different learning rate: blue corresponds to a learning rate of 0.001; orange to 0.005, and green to 0.05. The mean of the five repeated training runs is shown as a solid line, with the standard deviation shaded above and below. The sharp jumps are due to specific training runs terminating early due to the stopping criteria during training.

From the FEM solution we create a point cloud of training data by mapping the fluxes onto the nodes via the Clement interpolant, and then associating the nodal evaluations of the pressure and interpolated flux values with the scattered set of nodal positions.

Appendix C. Hyperparameter selection for battery problem

For the battery problem, we focus on the effects of varying the hyperparameters listed in Table C.6.

C.1. Learning rate

We explored optimal learning rates by performing a grid search. After a period of initial exploration, we found that the learning rates of $\{0.001, 0.005, 0.05\}$ warranted more study. Results with the smallest learning rate, of $\lambda = 0.001$, took significantly longer to train; for example the blue curves in Fig. C.12 lag behind the orange and green curves. While it is possible to train well with such a small learning rate, doing so takes substantially longer due to requiring more epochs to reach a comparable point of training accuracy. The other two learning rates of $\lambda = 0.05$ (orange), and $\lambda = 0.005$ (green), trained faster, and both provide comparable results. In Fig. C.12, we see the effects of increasing the number of fine-scale spline nodes on each of the terms in the overall loss. In all instances, training improves with greater resolution, as seen by the final loss values for the $L^2(\Omega)$ and $H^1_0(\Omega)$ components of the loss, which are plotted in Fig. C.13.

C.2. Number of fine-scale splines

The number of fine-scale splines was chosen to reflect a range of values that would instantiate models that would fit in GPU memory, given the size of the fine-to-coarse map of 1-forms and reflecting the final observations in Appendix A. The results of varying these are reflected in the refinement study in Section 6.1.2, and are complemented by the results in Fig. C.13. The results presented earlier are agnostic to the learning rate, assuming the problem trains to completion. For the smallest learning rate of $\lambda = 0.001$, this was not normally the case, but for the two other, larger learning rates, we matched the linear error rates of classical methods.

C.3. Quantity of interest

For the battery Problem, the relevant quantity of interest is the flux on the right side of the domain's boundary (i.e., the along the line $\{x = 1\}$). The relative error of this quantity of interest is plotted, for different fine-scale resolutions and various learning

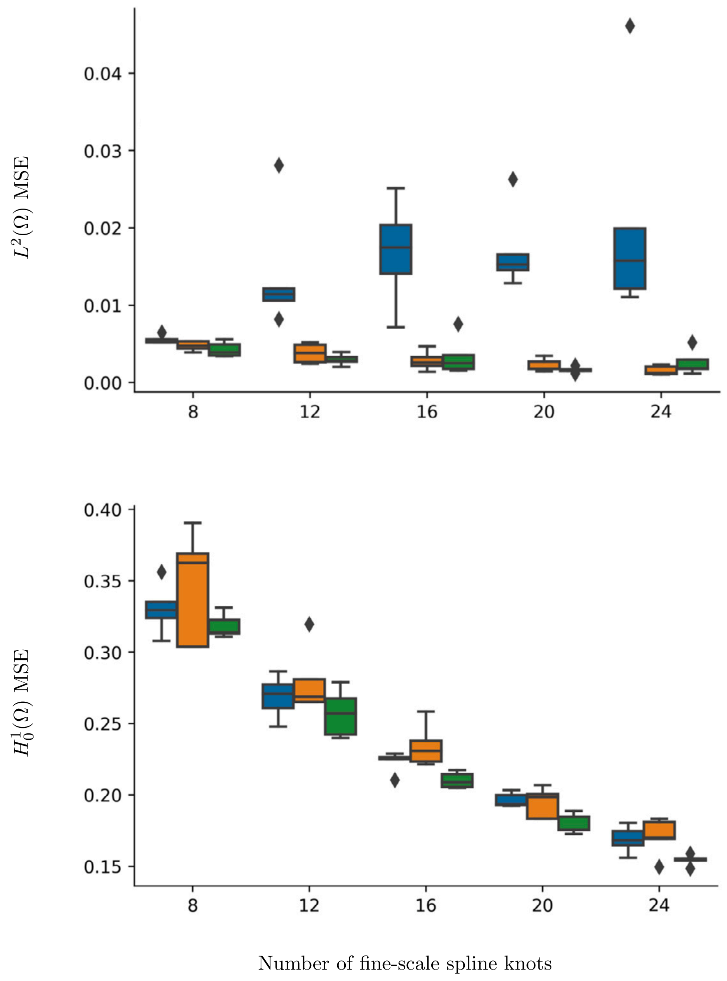


Fig. C.13. $L^2(\Omega)$ and $H_0^1(\Omega)$ loss terms for a range of fine-scale resolutions and learning rates. Each color represents a different learning rate: blue corresponds to a learning rate of 0.001; orange to 0.005, and green to 0.05. Box plots represent quartile data and diamond points mark outliers associated with five random initializations for training.

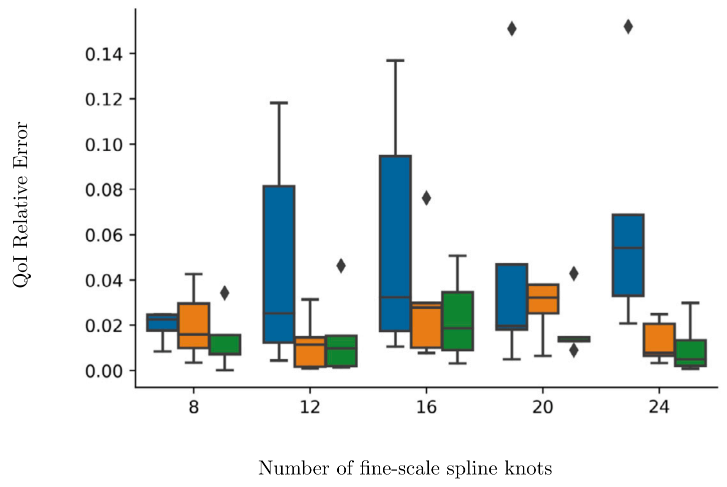


Fig. C.14. Relative error in quantity of interest for an increasing number of fine-scale spline nodes. Each color represents a different learning rate: blue corresponds to a learning rate of 0.001; orange to 0.005, and green to 0.05. For the two larger learning rates (0.005 and 0.5, respectively) the error remains relatively steady around 0.02, i.e., 2% error, with the best error occurring for a fine scale resolution of $n_x = 12$ and a learning rate of 0.05, for an error of 1%. Box plots represent quartile data (and the diamond points mark outliers) from using different random initializations for training.

rates, in Fig. C.14. Regardless of fine-scale resolution and learning rate, we achieve an average relative error of around 2% for the two larger learning rates, and a best relative error of 1%. Unlike with the loss function (discussed in Section 6.1.2), we do not see convergence with respect to the number of fine-scale splines. The observed plateau behavior is due to differences in the placement of the fine-scale splines as they try to resolve integrating the sharp discontinuities in the solution along the boundary, where the material in the battery changes between lithium, ion, or binder. That this quantity of interest is preserved is notable - there is no training data on the boundary of our domain, and there is no penalty to our loss function (as in PINNs [58,76] or Deep Ritz [81]) for achieving the flux correct along the boundary; the preservation of this quantity of interest comes from our proper treatment of the underlying chain complexes that preserve the correct physics, even as the size of the linear system has been reduced from 5.89 million degrees of freedom down to 136, corresponding to 8 POUs on the boundary and 8 POUs on the interior.

References

[1] Ralph Abraham, Jerrold E. Marsden, Tudor Ratiu, Manifolds, Tensor Analysis, and Applications, vol. 75, Springer Science & Business Media, 2012.
[2] James H. Adler, Casey Cavanaugh, Xiaozhe Hu, Ludmil T. Zikatanov, A finite-element framework for a mimetic finite-difference discretization of Maxwell's equations, SIAM J. Sci. Comput. 43 (4) (2021) A2638–A2659.
[3] Babak Maboudi Afkham, Jan S. Hesthaven, Structure preserving model reduction of parametric Hamiltonian systems, SIAM J. Sci. Comput. 39 (6) (2017) A2616–A2644.
[4] Martin S. Alnaes, Jan Blechta, Johan Hake, August Johansson, Benjamin Kehlet, Anders Logg, Chris Richardson, Johannes Ring, Marie E. Rognes, Garth N. Wells, The FEniCS project version 1.5, Arch. Numer. Softw. 3 (2015).
[5] Todd Arbogast, Gergina Pencheva, Mary F. Wheeler, Ivan Yotov, A multiscale mortar mixed finite element method, Multiscale Model. Simul. 6 (1) (2007) 319–346.
[6] Douglas N. Arnold, Finite Element Exterior Calculus, SIAM, 2018.
[7] Douglas N. Arnold, Pavel B. Bochev, Richard B. Lehoucq, Roy A. Nicolaides, Mikhail Shashkov, Compatible Spatial Discretizations, vol. 142, Springer Science & Business Media, 2007.
[8] Douglas N. Arnold, Richard S. Falk, Ragnar Winther, Finite element exterior calculus, homological techniques, and applications, Acta Numer. 15 (2006) 1–155.
[9] Vladimir Igorevich Arnol'd, Mathematical Methods of Classical Mechanics, vol. 60, Springer Science & Business Media, 2013.
[10] Ivo Babuška, Jens M. Melenk, The partition of unity method, Int. J. Numer. Methods Eng. 40 (4) (1997) 727–758.
[11] Randall Balestriero, Richard G. Baraniuk, Mad max: affine spline insights into deep learning, Proc. IEEE 109 (5) (2020) 704–727.
[12] Jacques Baranger, Jean-François Maitre, Fabienne Oudin, Connection between finite volume and mixed finite element methods, ESAIM: Math. Model. Numer. Anal. 30 (4) (1996) 445–465.
[13] Christine Bernardi, Yvon Maday, Anthony T. Patera, Domain decomposition by the mortar element method, in: Asymptotic and Numerical Methods for Partial Differential Equations with Critical Parameters, Springer, 1993, pp. 269–286.
[14] Miguel A. Bessa, R. Bostanabad, Zeliang Liu, A. Hu, Daniel W. Apley, C. Brinson, Wei Chen, Wing Kam Liu, A framework for data-driven analysis of materials under uncertainty: countering the curse of dimensionality, Comput. Methods Appl. Mech. Eng. 320 (2017) 633–667.
[15] Pavel B. Bochev, James M. Hyman, Principles of mimetic discretizations of differential operators, in: Compatible Spatial Discretizations, Springer, 2006, pp. 89–119.
[16] Alain Bossavit, Lauri Kettunen, Yee-like schemes on a tetrahedral mesh, with diagonal lumping, Int. J. Numer. Model. 12 (1–2) (1999) 129–142.
[17] Saifon Chaturantabut, Danny C. Sorensen, Nonlinear model reduction via discrete empirical interpolation, SIAM J. Sci. Comput. 32 (5) (2010) 2737–2764.
[18] Charles K. Chui, Multivariate Splines, SIAM, 1988.
[19] Eric C. Cyr, Mamikon A. Gulian, Ravi G. Patel, Mauro Perego, Nathaniel A. Trask, Robust training and initialization of deep neural networks: an adaptive basis viewpoint, in: Mathematical and Scientific Machine Learning, PMLR, 2020, pp. 512–536.
[20] Mathieu Desbrun, Anil N. Hirani, Melvin Leok, Jerrold E. Marsden, Discrete exterior calculus, arXiv preprint, arXiv:math/0508341, 2005.
[21] John Dolbow, Nicolas Moës, Ted Belytschko, Discontinuous enrichment in finite elements with a partition of unity method, Finite Elem. Anal. Des. 36 (3–4) (2000) 235–260.
[22] Charbel Farhat, Todd Chapman, Philip Avery, Structure-preserving, stability, and accuracy properties of the energy-conserving sampling and weighting method for the hyper reduction of nonlinear finite element dynamic models, Int. J. Numer. Methods Eng. 102 (5) (2015) 1077–1110.
[23] Charbel Farhat, Michel Lesoinne, Patrick LeTallec, Kendall Pierson, Daniel Rixen, Feti-dp: a dual–primal unified FETI method — Part I: A faster alternative to the two-level FETI method, Int. J. Numer. Methods Eng. 50 (7) (2001) 1523–1544.
[24] Carlos A.S. Ferreira, Teeratorn Kadeethum, Nikolaos Bouklas, Hamidreza M. Nick, A framework for upscaling and modelling fluid flow for discrete fractures using conditional generative adversarial networks, Adv. Water Resour. 166 (2022) 104264.
[25] Jacob Fish, Zheng Yuan, Multiscale enrichment based on partition of unity, Int. J. Numer. Methods Eng. 62 (10) (2005) 1341–1359.
[26] Michael S. Floater, Mean value coordinates, Comput. Aided Geom. Des. 20 (1) (2003) 19–27.
[27] Ari L. Frankel, Cosmin Safta, Coleman Alleman, Reese Jones, Mesh-based graph convolutional neural networks for modeling materials with microstructure, J. Mach. Learn. Model. Comput. 3 (1) (2022).
[28] Andrew Gillette, Alexander Rand, Chandrajit Bajaj, Construction of scalar and vector finite element families on polygonal and polyhedral meshes, Comput. Methods Appl. Math. 16 (4) (2016) 667–683.
[29] Chris Godsil, Gordon F. Royle, Algebraic Graph Theory, vol. 207, Springer Science & Business Media, 2001.
[30] Eric J. Hall, Søren Taverniers, Markos A. Katsoulakis, Daniel M. Tartakovsky, Ginns: graph-informed neural networks for multiscale physics, J. Comput. Phys. 433 (2021) 110192.
[31] Juncai He, Lin Li, Jinchao Xu, Chunyue Zheng, Relu deep neural networks and linear finite elements, arXiv preprint, arXiv:1807.03973, 2018.
[32] Emmanuel Hebey, Nonlinear Analysis on Manifolds: Sobolev Spaces and Inequalities, vol. 5, American Mathematical Soc., 2000.
[33] Joaquin Alberto Hernandez, Manuel Alejandro Caicedo, Alex Ferrer, Dimensional hyper-reduction of nonlinear finite element models via empirical cubature, Comput. Methods Appl. Mech. Eng. 313 (2017) 687–722.
[34] Cheng Huang, Karthik Duraisamy, Charles L. Merkle, Investigations and improvement of robustness of reduced-order models of reacting flow, AIAA J. 57 (12) (2019) 5377–5389.
[35] Thomas J.R. Hughes, Arif Masud, Jing Wan, A stabilized mixed discontinuous Galerkin method for Darcy flow, Comput. Methods Appl. Mech. Eng. 195 (25–28) (2006) 3347–3381.
[36] Ameya D. Jagtap, Ehsan Kharazmi, George Em Karniadakis, Conservative physics-informed neural networks on discrete domains for conservation laws: applications to forward and inverse problems, Comput. Methods Appl. Mech. Eng. 365 (2020) 113028.
[37] Teeratorn Kadeethum, Daniel O'Malley, Jan Niklas Fuhg, Youngsoo Choi, Jonghyun Lee, Hari S. Viswanathan, Nikolaos Bouklas, A framework for data-driven solution and parameter estimation of PDEs using conditional generative adversarial networks, Nat. Comput. Sci. 1 (12) (2021) 819–829.

[38] George Em Karniadakis, Ioannis G. Kevrekidis, Lu Lu, Paris Perdikaris, Sifan Wang, Liu Yang, Physics-informed machine learning, Nat. Rev. Phys. 3 (6) (2021) 422–440.
[39] Diederik P. Kingma, Jimmy Ba, Adam: a method for stochastic optimization, arXiv preprint, arXiv:1412.6980, 2014.
[40] Tamara G. Kolda, Brett W. Bader, Tensor decompositions and applications, SIAM Rev. 51 (3) (2009) 455–500.
[41] Tamara Gibson Kolda, Multilinear operators for higher-order decompositions, Technical report, Sandia National Laboratories (SNL), Albuquerque, NM, and Livermore, CA, 2006, SAND 2006-2081.
[42] Isaac E. Lagaris, Aristidis Likas, Dimitrios I. Fotiadis, Artificial neural networks for solving ordinary and partial differential equations, IEEE Trans. Neural Netw. 9 (5) (1998) 987–1000.
[43] Sanjay Lall, Petr Krysl, Jerrold E. Marsden, Structure-preserving model reduction for mechanical systems, Phys. D, Nonlinear Phenom. 184 (1–4) (2003) 304–318.
[44] Elisabeth Larsson, Victor Shcherbakov, Alfa Heryudono, A least squares radial basis function partition of unity method for solving PDEs, SIAM J. Sci. Comput. 39 (6) (2017) A2538–A2563.
[45] Kookjin Lee, Kevin T. Carlberg, Deep conservation: a latent-dynamics model for exact satisfaction of physical conservation laws, Proc. AAAI Conf. Artif. Intell. 35 (1) (2021) 277–285.
[46] Kookjin Lee, Nathaniel A. Trask, Ravi G. Patel, Mamikon A. Gulian, Eric C. Cyr, Partition of unity networks: deep hp-approximation, arXiv preprint, arXiv:2101.11256, 2021.
[47] Björn Liljegren-Sailer, Nicole Marheineke, On snapshot-based model reduction under compatibility conditions for a nonlinear flow problem on networks, J. Sci. Comput. 92 (2) (2022) 62.
[48] Jonni Lohi, Lauri Kettunen, Whitney forms and their extensions, J. Comput. Appl. Math. 393 (2021) 113520.
[49] Xiaoxin Lu, Dimitris G. Giovanis, Julien Yvonnet, Vissarion Papadopoulos, Fabrice Detrez, Jinbo Bai, A data-driven computational homogenization method based on neural networks for the nonlinear anisotropic electrical response of graphene/polymer nanocomposites, Comput. Mech. 64 (2) (2019) 307–321.
[50] Josiah Manson, Scott Schaefer, Moving least squares coordinates, Comput. Graph. Forum 29 (5) (2010) 1517–1524, Wiley Online Library.
[51] Ruairi M. Nestor, Mihai Basa, Martin Lastiwka, Nathan J. Quinlan, Extension of the finite volume particle method to viscous flow, J. Comput. Phys. 228 (5) (2009) 1733–1749.
[52] R. Nicolaides, D-Q. Wang, Convergence analysis of a covolume scheme for Maxwell's equations in three dimensions, Math. Comput. 67 (223) (1998) 947–963.
[53] Roy A. Nicolaides, Xiaonan Wu, Covolume solutions of three-dimensional div-curl equations, SIAM J. Numer. Anal. 34 (6) (1997) 2195–2203.
[54] Shaowu Pan, Karthik Duraisamy, Data-driven discovery of closure models, SIAM J. Appl. Dyn. Syst. 17 (4) (2018) 2381–2413.
[55] Junyoung Park, Jinkyoo Park, Physics-induced graph neural network: an application to wind-farm power estimation, Energy 187 (2019) 115883.
[56] Liqian Peng, Kamran Mohseni, Symplectic model reduction of Hamiltonian systems, SIAM J. Sci. Comput. 38 (1) (2016) A1–A27.
[57] Michael James David Powell, Approximation Theory and Methods, Cambridge University Press, 1981.
[58] Maziar Raissi, Paris Perdikaris, George E. Karniadakis, Physics-informed neural networks: a deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations, J. Comput. Phys. 378 (2019) 686–707.
[59] Chengping Rao, Yang Liu, Three-dimensional convolutional neural network (3D-CNN) for heterogeneous material homogenization, Comput. Mater. Sci. 184 (2020) 109850.
[60] Scott A. Roberts, Hector Mendoza, Victor E. Brunini, David R. Noble, A verified conformal decomposition finite element method for implicit, many-material geometries, J. Comput. Phys. 375 (2018) 352–367.
[61] Carmen Rodrigo, Francisco José Gaspar, Xiaozhe Hu, Ludmil Zikatanov, A finite element framework for some mimetic finite difference discretizations, Comput. Math. Appl. 70 (11) (2015) 2661–2673.
[62] Alexander Schein, Kevin T. Carlberg, Matthew J. Zahr, Preserving general physical properties in model reduction of dynamical systems via constrained-optimization projection, Int. J. Numer. Methods Eng. 122 (14) (2021) 3368–3399.
[63] Matthew W. Scroggs, Igor A. Baratta, Chris N. Richardson, Garth N. Wells, Basix: a runtime finite element basis evaluation library, J. Open Sour. Softw. 7 (73) (2022) 3982.
[64] Matthew W. Scroggs, Jorge S. Dokken, Chris N. Richardson, Garth N. Wells, Construction of arbitrary order finite element degree-of-freedom maps on polygonal and polyhedral cell meshes, ACM Trans. Math. Softw. 48 (2) (2022) 1–23.
[65] Khemraj Shukla, Mengjia Xu, Nathaniel Trask, George E. Karniadakis, Scalable algorithms for physics-informed neural and graph networks, Data-Cent. Eng. 3 (2022).
[66] Sierra Thermal Fluid Development Team, Sierra multimechanics module: Aria user manual - version 5.10, Technical Report SAND2022-12436, Sandia National Laboratories, 2022.
[67] Michael Spivak, Calculus on Manifolds: a Modern Approach to Classical Theorems of Advanced Calculus, CRC Press, 2018.
[68] Ishan Srivastava, Dan S. Bolintineanu, Jeremy B. Lechman, Scott A. Roberts, Controlling binder adhesion to impact electrode mesostructures and transport, ACS Appl. Mater. Interfaces 12 (31) (2020) 34919–34930.
[69] Aidan P. Thompson, H. Metin Aktulga, Richard Berger, Dan S. Bolintineanu, W. Michael Brown, Paul S. Crozier, Pieter J. in 't Veld, Axel Kohlmeyer, Stan G. Moore, Trung Dac Nguyen, Ray Shan, Mark J. Stevens, Julien Tranchida, Christian Trott, Steven J. Plimpton, LAMMPS - a flexible simulation tool for particle-based materials modeling at the atomic, meso, and continuum scales, Comput. Phys. Commun. 271 (2022) 108171.
[70] Nathaniel Trask, Amelia Henriksen, Carianne Martinez, Eric Cyr, Hierarchical partition of unity networks: fast multilevel training, Proc. Mach. Learn. Res. 145 (2022) 1–18.
[71] Nathaniel Trask, Andy Huang, Xiaozhe Hu, Enforcing exact physics in scientific machine learning: a data-driven exterior calculus on graphs, J. Comput. Phys. 456 (2022) 110969.
[72] Nathaniel Trask, Paul Kuberry, Compatible meshfree discretization of surface PDEs, Comput. Part. Mech. 7 (2) (2020) 271–277.
[73] The Trilinos Project Team, The Trilinos Project Website, https://trilinos.github.io, 2020. (Accessed 22 May 2020).
[74] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio, Yoshua Bengio, Graph attention networks, arXiv preprint, arXiv:1710.10903, 2017.
[75] Eugene L. Wachspress, A Rational Finite Element Basis, Academic Press, 1975.
[76] Sifan Wang, Yujun Teng, Paris Perdikaris, Understanding and mitigating gradient pathologies in physics-informed neural networks, arXiv preprint, arXiv:2001.04536, 2020.
[77] Sifan Wang, Yujun Teng, Paris Perdikaris, Understanding and mitigating gradient flow pathologies in physics-informed neural networks, SIAM J. Sci. Comput. 43 (5) (2021) A3055–A3081.
[78] Holger Wendland, Fast evaluation of radial basis functions: methods based on partition of unity, in: Approximation Theory X: Wavelets, Splines, and Applications, Citeseer, 2002.
[79] Holger Wendland, Scattered Data Approximation, vol. 17, Cambridge University Press, 2004.
[80] Wentao Yan, Stephen Lin, Orion L. Kafka, Yanping Lian, Cheng Yu, Zeliang Liu, Jinhui Yan, Sarah Wolff, Hao Wu, Ebot Ndip-Agbor, et al., Data-driven multi-scale multi-physics models to derive process–structure–property relationships for additive manufacturing, Comput. Mech. 61 (5) (2018) 521–541.
[81] Bing Yu, E. Weinan, The deep Ritz method: a deep learning-based numerical algorithm for solving variational problems, Commun. Math. Stat. 6 (1) (2018) 1–12.