# LEARNING INSIDE FINITE-VOLUME COMPUTATIONS WITH EFFECTIVE INTERFACE STATES

**Anonymous authors**
Paper under double-blind review

## ABSTRACT

Finite-volume methods provide a powerful framework for physical simulation by expressing conservation laws through local exchanges on a mesh. Building directly on their reconstruction–flux–update structure, we introduce the Finite-Volume Graph Neural Operator (FV-GNO), which learns the interface approximation within a prescribed numerical update rather than the full state transition. A graph network decodes effective two-sided interface states, while prescribed numerical fluxes, source treatment, and geometric assembly advance the solution. Trained end-to-end on state observations alone, these interfaces are optimized for prediction over the observation interval, without requiring interface or flux labels. The paired formulation inherits local balance for arbitrary network parameters, while a constructive analysis illustrates how retaining known nonlinear computation can provide a representation advantage beyond conservation alone. Experiments across diverse nonlinear equations on regular and irregular meshes demonstrate improved prediction over the evaluated neural baselines. On regular grids, FV-GNO also achieves higher accuracy and lower rollout cost than the tested second-order finite-volume schemes. Coarse-grid pretraining further enables accurate fine-grid prediction with few target trajectories.

## 1 INTRODUCTION

Classical numerical solvers provide accurate physical predictions through equation-based discretizations with well-studied stability and convergence properties (LeVeque, 2002). Their computational cost, however, can be substantial: high fidelity often requires fine meshes and small time steps (Kochkov et al., 2021). Neural operators and graph simulators offer a route to faster prediction by learning solution maps from data (Li et al., 2021; Pfaff et al., 2021; Kovachki et al., 2023; Sanchez-Gonzalez et al., 2020). Yet many such models replace the numerical update with a black-box predictor rather than retain its equation-specific computation. The network must then approximate not only unresolved dynamics, but also physical interactions and balance relations that the solver already knows how to compute.

We therefore learn the approximation within a finite-volume method, rather than the entire state transition. Equation-specific fluxes and sources, known physical coefficients, mesh geometry, and local balance prescribe how interface information contributes to the cell update. What cell averages do not directly provide is the interface information. Classical reconstruction supplies it through prescribed approximation rules; learning can instead adapt the approximation to the solution family and observation interval. This makes the finite-volume computation an inductive bias for prediction, not merely a constraint on its output.

Building on learned discretizations and conservative neural solvers (Horie & Mitsume, 2024; Lichtlé et al., 2026), we introduce the Finite-Volume Graph Neural Operator (FV-GNO). A graph network directly decodes two-sided interface states, while prescribed flux/source adapters and geometric assembly advance the cell states (Fig. 1). The interfaces are *effective*: they are optimized for the transition between observations, rather than required to reproduce instantaneous traces or a classical reconstruction. State-only supervision trains this approximation through the numerical update, without interface or flux labels. On regular control volumes, the paired assembly preserves local balance for any network parameters.



Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

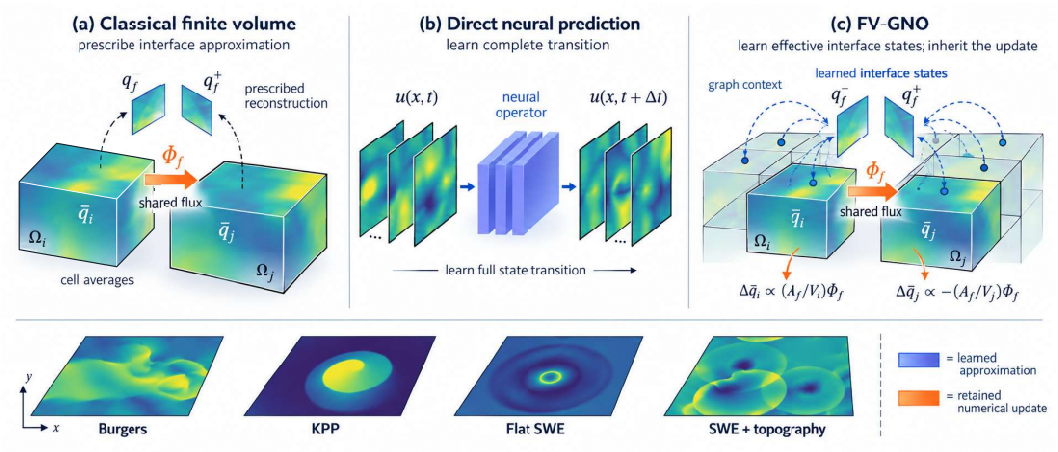


Figure 1: **Learn the interface approximation; inherit the update.** Classical finite volume prescribes the approximation; direct neural prediction learns the complete transition. FV-GNO learns effective interface states while retaining equation-specific interactions, geometric scaling, and local-balance assembly on regular control volumes. Blue denotes learned computation and orange retained numerical structure.

Across four regular-grid tasks and an irregular-mesh Euler extension, FV-GNO achieves lower mean rollout error than the evaluated external neural baselines. On the regular tasks, it also improves on the tested second-order finite-volume schemes in both accuracy and rollout cost, with speedups of 2.1–9.7× under a matched GPU protocol. On the homogeneous regular-grid tasks, the local finite-volume update remains more accurate than direct node prediction trained with a global conservation projection. This supports the numerical computation as an inductive bias beyond enforcing a conserved total. Moreover, pretraining at $64^2$ and adapting with only 16 trajectories at $128^2$ outperforms full-target-data scratch training on the three homogeneous tasks, demonstrating reuse of the learned interface approximation across resolutions.

Our contributions are:

- **Learning within a finite-volume computation.** We develop FV-GNO to learn the approximation used by a finite-volume update. Its primary realization directly decodes effective two-sided interface states and trains them through prescribed flux/source adapters and geometric assembly using state observations alone. The learned approximation is optimized for the observation-interval update while the equation-specific interaction remains prescribed.
- **Analysis of retained numerical structure.** We establish inherited local balance, construct an entropy-solution family demonstrating a representation advantage of known nonlinear computation over a specified capacity-limited direct class, and characterize an error floor induced by incompatible discrete invariants.
- **Prediction, efficiency, and cross-resolution reuse.** Neural and classical-solver comparisons assess prediction and accuracy–cost trade-offs. Coarse-grid pretraining improves fine-grid few-shot adaptation on disjoint trajectories. Local-organization and component controls explain the roles of interface learning, equation-specific interactions, and geometric assembly.

## 2 Related Work

**Representations for operator learning.** DeepONet (Lu et al., 2021) and FNO (Li et al., 2021) learn maps between function spaces. GINO (Li et al., 2023), GNOT (Hao et al., 2023), and Transolver (Wu et al., 2024) address complex geometries. CNO (Raonic et al., 2023) and F-FNO (Tran et al., 2021) develop convolutional and factorized spectral operator representations. PDE-Refiner (Lippe et al., 2023) studies low-amplitude modes; DPOT (Hao et al., 2024) and Poseidon (Herde et al., 2024) use

multi-equation pretraining. Our focus is the numerical organization in which a learned representation is used, rather than a new pretraining strategy.

**Learning inside numerical algorithms.** MeshGraphNets and MP-PDE (Pfaff et al., 2021; Brandstetter et al., 2022) learn mesh interactions; (U)NFV (Lichtlé et al., 2026) predicts numerical fluxes. FLUXGNN (Horie & Mitsume, 2024) combines finite-volume geometry, encoded-space flux passing, and learned interpolation. FV-GNO directly decodes effective two-sided states, rather than interpolation weights, and optimizes them through prescribed flux/source adapters for the observation-interval update. This separates the learned interface approximation from the known equation-specific computation and mesh-dependent assembly.

**Known computation and hard conservation.** Known-operator learning studies approximation bounds with prescribed computational components (Maier et al., 2019); conservation-law-encoded operators impose selected identities architecturally (Liu et al., 2024). Our theory separates these roles: known nonlinear computation can provide a representation benefit beyond a conserved total, while an incompatible invariant can exclude the target. Our Trixi suite complements PDEBench (Takamoto et al., 2022) with finite-volume interventions and geometry-resolved diagnostics.

## 3 METHOD

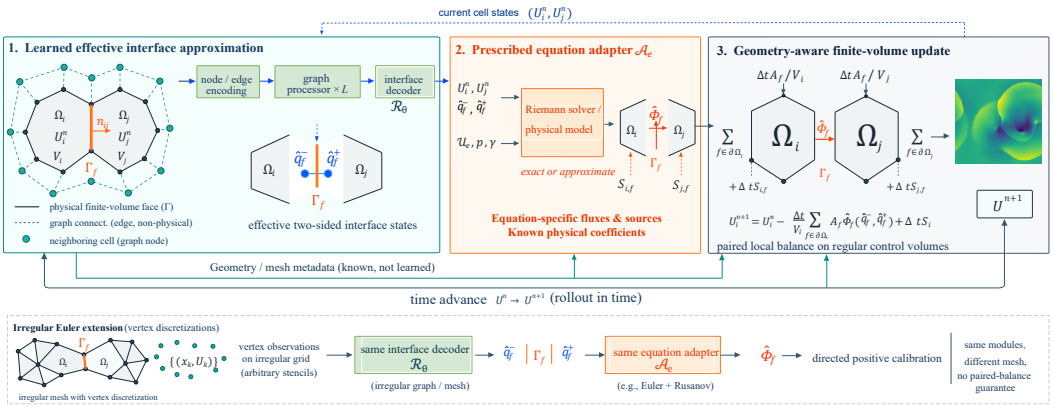


Figure 2: FV-GNO inherits the finite-volume organization of a state transition. A graph processor learns effective interface states; an equation adapter evaluates their numerical fluxes and sources; known geometry weights regular-grid exchanges in a paired balance update. The processor and equation adapter are replaceable.

### 3.1 FINITE-VOLUME STRUCTURE AND LEARNED APPROXIMATION

Consider a balance law for the conservative state $U$,

$$\partial_t U + \nabla \cdot F(U) = S(U, x). \tag{1}$$

On a fixed regular mesh, cell $i$ has volume $V_i > 0$ and average state $U_i$. Classical finite-volume schemes approximate its integral balance using reconstructed interface states, numerical fluxes, and time integration (Appendix A.1). FV-GNO learns effective interface states while retaining the subsequent numerical computation (Fig. 2):

$$Z_\theta = R_\theta(U^n, \mathcal{G}), \qquad U^{n+1} = U^n + \mathcal{P}_{\mathcal{G},\Delta t}(U^n, Z_\theta). \tag{2}$$

Here $\mathcal{G}$ provides connectivity, geometry, and static fields; $Z_\theta$ contains two states per face. The prescribed map $\mathcal{P}$ evaluates equation-specific fluxes and sources, weights them by known geometry, and assembles the cell update. One model update advances the solution over the saved observation interval $\Delta t$.

### 3.2 GRAPH-BASED INTERFACE RECONSTRUCTION

The graph has one node per cell and an exchange edge per mesh face. Node encoders embed current states and static fields; edge encoders embed displacement, normals, and geometric measures. The processor aggregates neighboring features with kernels conditioned on endpoint embeddings, their difference, and encoded geometry. After $L$ layers, an edge decoder predicts states on the $i$- and $j$-sides of each face $f = (i, j)$:

$$(\widehat{q}_f^{-}, \widehat{q}_f^{+}) = D_\theta(z_i^L, z_j^L, e_{ij}),$$

where $z_i^L$ is the processed node embedding and $e_{ij}$ the encoded edge geometry. Each interface prediction therefore uses graph context beyond its incident cells (Battaglia et al., 2018). Additional irregular-mesh neighbors provide context but do not add physical exchange faces.

The processor can change without altering the numerical update or its regular-grid conservation structure. We test this flexibility by replacing it with a message-passing neural network (Gilmer et al., 2017), retaining the interface decoder and numerical modules. Appendix B.3 specifies the processor and decoder.

### 3.3 NUMERICAL INTERACTIONS AND CONSERVATIVE ASSEMBLY

Orient each interior face $f = (i, j)$ by a unit normal $n_f$ from $i$ to $j$, and use outward normals on external boundaries. Let $A_f$ denote the face measure. The equation adapter converts $\widehat{q}_f^{\pm}$ to conservative coordinates $\widehat{U}_f^{\pm}$ when necessary and evaluates a numerical flux. For example, local Lax–Friedrichs uses

$$\widehat{\Phi}_f^{\mathrm{LLF}} = \frac{F_{n_f}(\widehat{U}_f^{-}) + F_{n_f}(\widehat{U}_f^{+})}{2} - \frac{\alpha_f}{2}(\widehat{U}_f^{+} - \widehat{U}_f^{-}),$$

where $F_{n_f}(U) = F(U) \cdot n_f$ and $\alpha_f$ is the equation-specific wave-speed estimate. Burgers, KPP, and Euler use this flux family; Burgers additionally uses centered viscous exchange from current cell states. Shallow water uses HLL (Harten et al., 1983), with hydrostatic reconstruction and bed-source corrections for topography (Appendix B.2). Physical coefficients and static fields remain prescribed. Each shared transport flux $\widehat{\Phi}_f$ is evaluated once and reused by both cells. Side-specific source corrections are included once in $\widehat{S}_i$, separately from transport.

Let $\partial i$ contain the faces of cell $i$, with $s_{if} = +1$ when $n_f$ points outward and $-1$ otherwise. The regular-grid update is

$$\boxed{U_i^{n+1} = U_i^n - \frac{\Delta t}{V_i} \sum_{f \in \partial i} A_f s_{if} \widehat{\Phi}_f + \Delta t\, \widehat{S}_i.} \tag{3}$$

The factor $\Delta t A_f$ converts a flux into transferred content; division by $V_i$ gives the cell-average change. Regular-grid models retain this coefficient, with the additional calibration multiplier fixed to one. Appendix D examines the interface, adapter, and assembly through component ablations.

**Conservation through paired exchange.** For an interior face $f = (i, j)$, the same flux supplies the two transport contributions

$$(\Delta U_i)_f = -\frac{\Delta t A_f}{V_i} \widehat{\Phi}_f, \qquad (\Delta U_j)_f = +\frac{\Delta t A_f}{V_j} \widehat{\Phi}_f.$$

Thus $V_i(\Delta U_i)_f + V_j(\Delta U_j)_f = 0$ for all network parameters. Periodic partners are treated as interior faces. Multiplying Eq. (3) by $V_i$ and summing over any union of cells therefore leaves only boundary exchanges and sources. On the full domain,

$$\sum_i V_i(U_i^{n+1} - U_i^n) = -\Delta t \sum_{f \in \partial\Omega} A_f \widehat{\Phi}_f + \Delta t \sum_i V_i \widehat{S}_i. \tag{4}$$

The source-free periodic tasks conserve the volume-weighted total of every conservative channel; topographic shallow water conserves mass, while the bed source changes momentum. This identity holds in exact arithmetic, without a conservation penalty or projection. Appendix A.1 proves it, and Appendix E.1 audits executed updates and output conversions.

**Irregular-mesh extension.** Euler observations are vertex values, not averages over reconstructed dual cells. We retain the interface and equation modules but allow positive directed calibration of the update coefficients. Directed calibration need not preserve equal-and-opposite volume-weighted exchanges, so the regular-grid identity is not imposed. Appendix E.2 compares dual geometry, point estimates, calibration, and forced pairing.

**Efficient graph execution.** Cached geometry, fixed-degree reductions, folded endpoint/difference projections, and removal of redundant graph copies accelerate the same paired computation. Appendix B.5 gives the optimized execution and deployment contract.

### 3.4 Observation-interval learning and autoregressive prediction

**Supervision through the complete numerical update.** For a training input, the graph network reconstructs the two interface states, the adapter evaluates their transport and source terms, and Eq. (3) advances the conservative state. The output is then converted to the stored observation variables. Writing $\widehat{q}_\theta$ for this prediction, $q^{\mathrm{obs}}$ for its target, and $\mathcal{N}$ for the training-data normalizer, the state loss is

$$\mathcal{L}(\theta) = \mathrm{MSE}\big(\mathcal{N}(\widehat{q}_\theta), \mathcal{N}(q^{\mathrm{obs}})\big)\,. \tag{5}$$

The comparison is made after the numerical update, so the interface decoder is trained for its effect on the next observed state. Gradients pass through the prescribed adapter and geometric assembly into the decoder and processor; the equation coefficients, flux formulas, and mesh measures remain known inputs. No intermediate reconstruction or flux targets are needed.

**Learning over the saved time interval.** The reference solver may take multiple internal steps between two observations, whereas one learned update spans their full interval $\Delta t$. The effective states consequently serve as inputs to this finite-time computation, rather than labels for an instantaneous reconstruction. On the regular tasks, 20 one-step epochs establish this state-to-state approximation, followed by four epochs of three-step autoregressive training. The latter exposes the reconstruction to predicted intermediate states and trains it through successive applications of the same numerical update. Appendix B.3 gives the optimizer, normalization, and task-specific settings.

**Rollout with reusable local maps.** Prediction starts from a single observed state. At each subsequent step, the current predicted state is encoded, new effective interfaces are reconstructed, and the equation adapter and assembly advance the state again. Static mesh quantities are cached; state-dependent interactions are recomputed. The same local encoders, processor kernels, and decoder act on every mesh entity with shared parameters. Changing the number of cells therefore changes the graph and its geometric coefficients, rather than the dimensions of these learned maps. This separation provides the computational organization used in the coarse-to-fine adaptation experiment in Section 5.4.

## 4 Theoretical Analysis

The paired update preserves local balance; retaining known nonlinear computation can additionally reduce representation complexity. However, imposed discrete relations must admit the target. Full assumptions and proofs are in Appendix A.

**Structure survives learned approximation.** The paired update balances every union of cells for arbitrary reconstructions. Exact PDE averages obey the same integral structure; learning can change the approximation without relaxing the balance (Appendix A.1).

**Known computation can reduce representation complexity.**

**Proposition 4.1** (A constructive approximation advantage)**.** *For a smooth, strictly convex scalar flux, there is a family of periodic entropy solutions with a shock and a rarefaction, a nondegenerate amplitude interval, and a positive time window before waves cross cell faces, such that a fixed local linear reconstruction followed by any consistent numerical flux gives the exact cell-average increment. Under an amplitude sampling density bounded below, any complete direct prediction class that is*

*piecewise affine in amplitude with at most $P$ pieces has best population $L^2$ error at least $c\tau/P^2 > 0$ at prediction time $\tau > 0$. The bound also holds for its conservative subclass.*

The constant $c > 0$ depends on the amplitude law, mesh, and flux curvature. The prescribed nonlinear flux carries a part of the target map that the limited direct class must approximate. The result isolates this representation mechanism for the stated function classes; Appendix A.2 gives the higher-order extension and its scope.

**Discrete relations must admit the target.** Let $m$ be a scalar-channel target increment, let $M = \mathrm{diag}(V_i)$ define its squared error norm, and suppose an alternative closed-domain assembly uses positive volumes $\widetilde{v}_i > 0$ and shared face exchanges, so $\widetilde{v}^{\mathsf{T}}\widehat{m} = 0$. Then

$$\|\widehat{m} - m\|_M^2 \;\geq\; \frac{|\widetilde{v}^{\mathsf{T}} m|^2}{\widetilde{v}^{\mathsf{T}} M^{-1} \widetilde{v}}. \tag{6}$$

On a connected graph with unrestricted shared fluxes, the bound is attained. An incompatible invariant can therefore create an irreducible error floor. Appendices A.3–A.4 give the proof and finite-calibration analysis. The Euler controls examine compatibility empirically for open-boundary vertex observations, outside this closed-domain setting.

# 5 Experiments

## 5.1 Experimental setup

We construct five datasets with Trixi.jl (Ranocha et al., 2022). Four periodic tasks—vector viscous Burgers, nonconvex KPP, flat shallow water, and shallow water over random topography—provide $128^2$ observations derived from conservative cell averages. Compressible Euler provides vertex observations on irregular bump-channel meshes with prescribed boundaries. Shock-capturing DG reference calculations with polynomial degrees three or four resolve each saved transition through internal time steps; the learned model predicts the same transition in a single update. Train/test splits are 270/30 for Burgers and topography, 280/20 for KPP and flat water, and 300/20 for Euler.

The default model uses a ten-block processor with approximately 1.2 million parameters. The main prediction benchmark uses three independently initialized training seeds (123/456/789) on every task, including Euler. A single input state initializes 79 rollout predictions, or 50 for topography. The evaluated FNO, U-Net, and DPOT configurations use five input states and are scored on the remaining 75 frames, or 46 for topography.

Macro-RMSE averages the space–time RMSE of each trajectory–channel pair in physical units. Conservation audits use physical volume weights and conservative states. Appendix B provides the equations, data generators, observation maps, and training and numerical protocols.

## 5.2 Prediction with the finite-volume architecture

**One numerical architecture across five tasks.** FV-GNO achieves the lowest reported macro-RMSE among the evaluated external neural methods on each regular task (Table 1). Relative to the one-state FLUXGNN baseline, the three-seed mean error is lower by approximately 69%, 81%, and 60% on the three homogeneous tasks. Appendices C.3 and G report high-jump-region accuracy and rollout fields.

The irregular Euler extension reaches 0.1136 mean macro-RMSE over three training seeds, compared with 0.128 for GNOT and 0.493 for Transolver (Table 2). The same interface–flux decomposition therefore operates on both regular cell graphs and irregular vertex graphs, with geometry adapted to the observation representation. On reference-selected high-pressure-jump regions, it also resolves localized dynamics more accurately than these baselines (Appendix C.3). In the Euler front diagnostic, the top 10% of nodes are selected by their reference pressure-jump scores. Region macro-RMSE is 0.293 for FV-GNO, compared with 0.394 for GNOT and 0.906 for Transolver. Regular-grid diagnostics likewise evaluate reference-selected jump regions, alongside final-time profiles and the full rollout fields. These measurements examine the localized fronts that motivate a learned interface approximation, in addition to the domain-wide averages used in the main tables.

Table 1: Regular-grid macro-RMSE ($\times 10^{-3}$; lower is better). Entries are means over three independent training seeds. One-state models score from the initial frame; five-state models score after context. A dash denotes an unstable rollout. Component ablations are reported in Appendix D.

| Method | States | Burgers | KPP | Flat SWE | SWE + topo. |
|---|---|---|---|---|---|
| FV-GNO | 1 | **12.280** | **35.413** | **11.676** | **4.402** |
| FLUXGNN | 1 | 40.013 | 189.955 | 29.156 | - |
| DPOT | 5 | 33.202 | 144.522 | 18.840 | 23.127 |
| FNO | 5 | 479.207 | 339.753 | 41.317 | 27.859 |
| U-Net | 5 | 62.236 | 155.025 | 80.051 | 55.419 |

Table 2: Irregular Euler macro-RMSE, mean and sample SD over three training seeds. FV-GNO retains directed calibration. All methods receive one initial state and the prescribed boundary sequence. Component ablations are reported in Appendix D.

| Method | States | Mean | SD |
|---|---|---|---|
| FV-GNO (calibrated) | 1 | 0.114 | 0.015 |
| GNOT | 1 | 0.128 | 0.004 |
| Transolver | 1 | 0.493 | 0.062 |

### 5.3 Does learning improve prescribed numerical approximations?

Table 3: Accuracy and matched GPU rollout cost: RTX 5090, FP32 IEEE, eager, batch one. Times average per-run medians. DG2 has no GPU time. Appendix B.6 specifies the protocol.

| Method | Burgers | | KPP | | Flat SWE | | SWE + topo. | |
|---|---|---|---|---|---|---|---|---|
| | RMSE | Time (s) | RMSE | Time (s) | RMSE | Time (s) | RMSE | Time (s) |
| FV-GNO | 0.012 | 0.675 | 0.035 | 0.661 | 0.012 | 0.741 | 0.004 | 0.497 |
| FV1 | 0.043 | 0.264 | 0.445 | 0.232 | 0.044 | 0.425 | 0.033 | 0.838 |
| FV2–MC | 0.016 | 1.712 | 0.489 | 1.372 | 0.013 | 3.074 | 0.006 | 4.811 |
| FV2–MM1 | 0.019 | 1.473 | 0.187 | 2.147 | 0.021 | 2.709 | 0.010 | 4.293 |
| DG2 | 0.010 | - | 0.081 | - | 0.007 | - | 0.004 | - |

We compare first-order finite volume (FV1), MUSCL reconstructions with MC and minmod limiters (FV2-MC/FV2-MM1), and degree-two DG (DG2). Classical schemes use CFL-limited substeps; FV-GNO learns one transition at the observation interval.

**Retain the balance, improve the approximation.** FV-GNO has lower reference error than FV1 and both FV2 reconstructions on all four tasks (Table 3; Fig. 3). Under the common GPU protocol, the learned update is 2.1–9.7 times faster than the two FV2 schemes. Against FV1, topographic water is also faster (0.497 s versus 0.838 s per rollout); other tasks trade runtime for accuracy. DG2 has lower error on Burgers and both water tasks; KPP favors FV-GNO. DG2 has no comparable GPU timing.

### 5.4 Cross-resolution few-shot adaptation

An interface approximation learned on a coarse graph can initialize prediction on a finer graph: the network supplies local states, while the target mesh supplies its own face measures and cell volumes. We examine this capability by pretraining FV-GNO at $64^2$ and fine-tuning the same model architecture at $128^2$. Both resolutions retain the prescribed equation adapters, fixed finite-volume geometry, and paired assembly. The target graph has four times as many cells, but the learned local maps and their parameter dimensions remain unchanged.

**Transfer across disjoint trajectories.** Source pretraining, target adaptation, and target testing use disjoint trajectory splits A, B, and C. Their sizes are 135/135/30 for Burgers and 140/140/20 for KPP and flat shallow water. We focus on 16 target trajectories drawn from split B. Pretrained and scratch

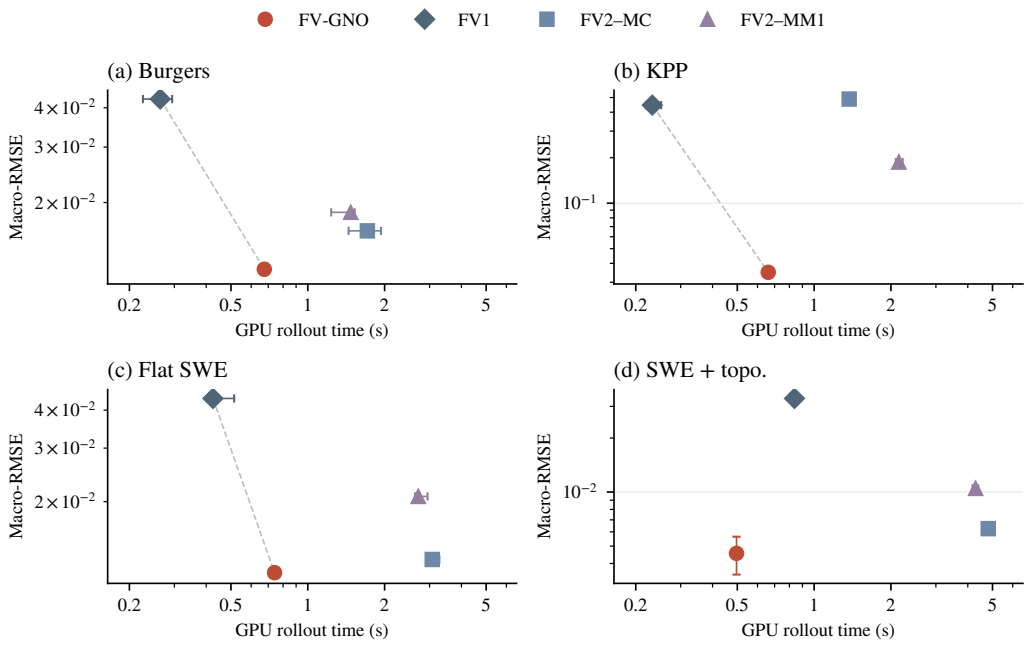


Figure 3: **Learning changes the finite-volume accuracy–cost trade-off.** All methods use the same resident GPU timing protocol; dashed lines connect nondominated measured configurations. Horizontal bars span mean within-run p10–p90 costs, including trajectory variation; vertical bars show training-run sample SD. Lower and left are better. Table 3 gives the values.

models use the same target examples, frozen source normalization, and target optimizer steps; all target evaluations use split C. Thus, transfer is assessed on new trajectories at the finer resolution, not on finer samples of the source training trajectories. Appendix F.1 gives the complete target-data sweep and the topographic results.

Table 4: $64^2 \to 128^2$ few-shot adaptation. Macro-RMSE is evaluated on disjoint target test trajectories. Scratch and pretrained 16-shot models receive identical target data and optimizer steps. Full-split scratch uses 135 target trajectories for Burgers and 140 for KPP and flat SWE. Reduction compares the two 16-shot models, using the recorded pretrained/scratch ratios. The full sweep is in Table 22.

| Task | Scratch, 16 | Pretrained, 16 | Reduction (%) | Scratch, full |
|---|---|---|---|---|
| Burgers | 0.046 | **0.017** | 62.8 | 0.028 |
| KPP | 0.146 | **0.049** | 66.1 | 0.105 |
| Flat SWE | 0.043 | **0.016** | 63.5 | 0.022 |

**A reusable interface representation.** Coarse-grid pretraining lowers the 16-shot target error by 62.8%, 66.1%, and 63.5% on Burgers, KPP, and flat shallow water, respectively (Table 4). The adapted model also outperforms a scratch model trained on the full target split in all three tasks. For example, KPP reaches 0.049 with 16 target trajectories, compared with 0.105 using all 140 target trajectories from scratch. The benefit therefore extends beyond fitting the source discretization: the learned local approximation provides an effective initialization for different cell sizes and unseen target dynamics.

**Coarse pretraining, limited fine-grid data.** The 16-shot setting uses only 11.4–11.9% of the full target-training split, a reduction of approximately 88–89% in target trajectories. Moreover, the source graph contains 4,096 cells rather than 16,384. With the fixed-degree local processor, coarse pretraining operates on correspondingly smaller collections of node and face features. This organizes learning around a smaller spatial workload before fine-grid adaptation, while preserving the

equation-specific computation at both stages. The result is a practical route to reducing dependence on high-resolution training data: learn reusable interface approximations on the coarse grid, then adapt them using a small fine-grid sample.

### 5.5 Local numerical organization beyond a conserved total

A global invariant alone does not determine how neighboring cells exchange conserved quantities. We compare FV-GNO with a direct nodal predictor and a nodal predictor whose volume-weighted global conservation projection is included during training. The three models share the graph-processor family. On all three homogeneous tasks, FV-GNO has lower three-seed mean error than either nodal control (Fig. 4). This supports learning within a local finite-volume computation rather than learning a full state transition followed by a global correction.

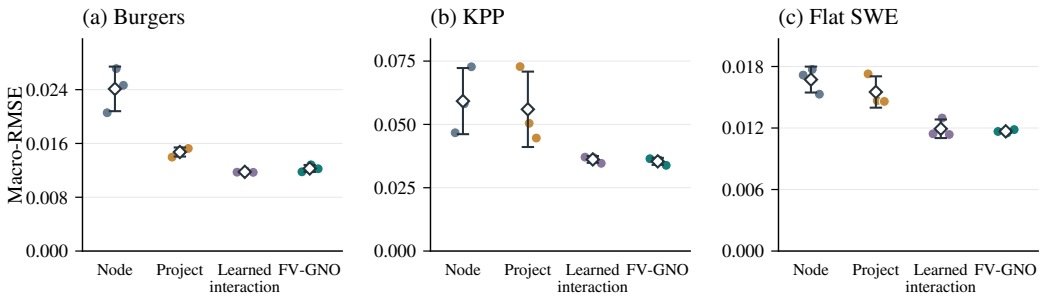


Figure 4: **Local numerical organization provides more than a correct total.** Nodal prediction and its training-time global conservation projection are compared with FV-GNO. Points are three training runs; diamonds show means and sample SD. The additional learned-interaction control uses the same local assembly and is defined in Appendix D.4.

For Burgers, global projection lowers the nodal mean error from approximately 0.024 to 0.015, whereas FV-GNO reaches 0.012. For KPP, the corresponding values are 0.059, 0.056, and 0.035. Local assembly therefore contributes predictive structure beyond correcting aggregate drift. Every shared face exchange determines compatible increments for two neighbors, and the cancellation extends to every union of cells, not only the full domain. The network learns the interface information needed by these exchanges while the numerical assembly determines how that information advances the field.

The component studies in Appendix D then locate the source of these improvements. Replacing learned interfaces with incident cell states while holding the numerical update fixed increases reference-state one-step error on every regular task. Geometry and pairing controls examine how the reconstructed information is assembled, and long-horizon equilibrium tests evaluate the role of the prescribed physical interaction. These interventions separate interface approximation, equation evaluation, and local balance instead of treating conservation as a single output property.

## 6 Conclusion

FV-GNO learns effective interface approximations inside a finite-volume computation, retaining prescribed physical interactions, mesh geometry, and regular-grid local balance. The model predicts diverse regular- and irregular-mesh dynamics accurately; regular-grid experiments further establish favorable accuracy–cost trade-offs and coarse-to-fine reuse. Pretraining on a coarse grid enables accurate fine-grid prediction with limited target data; local-organization controls further distinguish this design from imposing a conserved total on a direct predictor. The central principle is to learn the unresolved approximation while allowing the numerical method to perform the known physical computation.

### AI USE STATEMENT

The authors led the research, including problem formulation, methodological choices, experimental design, and the main manuscript draft. Generative AI was used as an assistive tool to refine research ideas and execution plans, develop code from author-specified instructions, polish author-written text, and draft selected passages. For the mathematical analysis, the authors provided the ideas and proof directions, and AI assisted in developing intermediate steps and organizing detailed arguments. The authors retained control over substantive decisions and the final presentation, and take responsibility for the code, experimental results, mathematical claims, references, and manuscript.

### REPRODUCIBILITY STATEMENT

Sections 3–5 describe the model and evaluation protocol. Appendices A–G provide proofs, dataset and training protocols, per-run results, and diagnostics. The LaTeX source package rebuilds the manuscript from fixed tables and PDF figures. Reproducing the experiments additionally requires the experimental code, datasets, and trained weights. Evaluation and artifact provenance are described in Appendix B.

## REFERENCES

Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. Layer normalization. *arXiv preprint arXiv:1607.06450*, 2016.

F. Bassi and S. Rebay. A high-order accurate discontinuous finite element method for the numerical solution of the compressible navier–stokes equations. *Journal of Computational Physics*, 131(2): 267–279, 1997. ISSN 0021-9991. doi: https://doi.org/10.1006/jcph.1996.5572. URL https://www.sciencedirect.com/science/article/pii/S0021999196955722.

Peter W Battaglia, Jessica B Hamrick, Victor Bapst, Alvaro Sanchez-Gonzalez, Vinicius Zambaldi, Mateusz Malinowski, Andrea Tacchetti, David Raposo, Adam Santoro, Ryan Faulkner, et al. Relational inductive biases, deep learning, and graph networks. *arXiv preprint arXiv:1806.01261*, 2018.

Chris M. Bishop. Training with noise is equivalent to tikhonov regularization. *Neural Computation*, 7(1):108–116, 01 1995. ISSN 0899-7667. doi: 10.1162/neco.1995.7.1.108. URL https://doi.org/10.1162/neco.1995.7.1.108.

Johannes Brandstetter, Daniel E. Worrall, and Max Welling. Message passing neural PDE solvers. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=vSix3HPYKSU.

Carsten Burstedde, Lucas C. Wilcox, and Omar Ghattas. p4est: Scalable algorithms for parallel adaptive mesh refinement on forests of octrees. *SIAM Journal on Scientific Computing*, 33(3):1103–1133, 2011. doi: 10.1137/100791634. URL https://doi.org/10.1137/100791634.

Guoxian Chen and Sebastian Noelle. A new hydrostatic reconstruction scheme based on subcell reconstructions. *SIAM Journal on Numerical Analysis*, 55(2):758–784, 2017. doi: 10.1137/15M1053074. URL https://doi.org/10.1137/15M1053074.

Justin Gilmer, Samuel S. Schoenholz, Patrick F. Riley, Oriol Vinyals, and George E. Dahl. Neural message passing for quantum chemistry. In Doina Precup and Yee Whye Teh (eds.), *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pp. 1263–1272. PMLR, 06–11 Aug 2017. URL https://proceedings.mlr.press/v70/gilmer17a.html.

Zhongkai Hao, Zhengyi Wang, Hang Su, Chengyang Ying, Yinpeng Dong, Songming Liu, Ze Cheng, Jian Song, and Jun Zhu. GNOT: A general neural operator transformer for operator learning. In Andreas Krause, Emma Brunskill, Kyunghyun Cho, Barbara Engelhardt, Sivan Sabato, and Jonathan Scarlett (eds.), *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pp. 12556–12569. PMLR, 23–29 Jul 2023. URL https://proceedings.mlr.press/v202/hao23c.html.

Zhongkai Hao, Chang Su, Songming Liu, Julius Berner, Chengyang Ying, Hang Su, Anima Anandkumar, Jian Song, and Jun Zhu. DPOT: Auto-regressive denoising operator transformer for large-scale PDE pre-training. In Ruslan Salakhutdinov, Zico Kolter, Katherine Heller, Adrian Weller, Nuria Oliver, Jonathan Scarlett, and Felix Berkenkamp (eds.), *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 17616–17635. PMLR, 21–27 Jul 2024. URL https://proceedings.mlr.press/v235/hao24d.html.

Amiram Harten, Peter D. Lax, and Bram van Leer. On upstream differencing and godunov-type schemes for hyperbolic conservation laws. *SIAM Review*, 25(1):35–61, 1983. doi: 10.1137/1025002. URL https://doi.org/10.1137/1025002.

Sebastian Hennemann, Andrés M. Rueda-Ramírez, Florian J. Hindenlang, and Gregor J. Gassner. A provably entropy stable subcell shock capturing approach for high order split form dg for the compressible euler equations. *Journal of Computational Physics*, 426:109935, 2021. ISSN 0021-9991. doi: https://doi.org/10.1016/j.jcp.2020.109935. URL https://www.sciencedirect.com/science/article/pii/S0021999120307099.

Maximilian Herde, Bogdan Raonić, Tobias Rohner, Roger Käppeli, Roberto Molinaro, Emmanuel de Bézenac, and Siddhartha Mishra. Poseidon: Efficient foundation models for pdes. In A. Globerson, L. Mackey, D. Belgrave, A. Fan, U. Paquet, J. Tomczak, and C. Zhang (eds.), *Advances in Neural Information Processing Systems*, volume 37, pp. 72525–72624. Curran Associates, Inc., 2024. doi: 10.52202/079017-2311. URL https://proceedings.neurips.cc/paper_files/paper/2024/file/84e1b1ec17bb11c57234e96433022a9a-Paper-Conference.pdf.

Masanobu Horie and Naoto Mitsume. Graph neural PDE solvers with conservation and similarity-equivariance. In Ruslan Salakhutdinov, Zico Kolter, Katherine Heller, Adrian Weller, Nuria Oliver, Jonathan Scarlett, and Felix Berkenkamp (eds.), *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 18785–18814. PMLR, 21–27 Jul 2024. URL https://proceedings.mlr.press/v235/horie24a.html.

David I. Ketcheson, Randall J. LeVeque, and Mauricio J. del Razo. *Riemann Problems and Jupyter Solutions*. SIAM, 2020.

Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

Dmitrii Kochkov, Jamie A. Smith, Ayya Alieva, Qing Wang, Michael P. Brenner, and Stephan Hoyer. Machine learning-accelerated computational fluid dynamics. *Proceedings of the National Academy of Sciences*, 118(21):e2101784118, 2021. doi: 10.1073/pnas.2101784118.

Nikola Kovachki, Zongyi Li, Burigede Liu, Kamyar Azizzadenesheli, Kaushik Bhattacharya, Andrew Stuart, and Anima Anandkumar. Neural operator: Learning maps between function spaces with applications to pdes. *Journal of Machine Learning Research*, 24(89):1–97, 2023. URL http://jmlr.org/papers/v24/21-1524.html.

Alexander Kurganov, Guergana Petrova, and Bojan Popov. Adaptive semidiscrete central-upwind schemes for nonconvex hyperbolic conservation laws. *SIAM Journal on Scientific Computing*, 29 (6):2381–2401, 2007. doi: 10.1137/040614189. URL https://doi.org/10.1137/040614189.

Randall J. LeVeque. *Finite Volume Methods for Hyperbolic Problems*. Cambridge Texts in Applied Mathematics. Cambridge University Press, 2002. doi: 10.1017/CBO9780511791253.

Zongyi Li, Nikola Borislavov Kovachki, Kamyar Azizzadenesheli, Burigede liu, Kaushik Bhattacharya, Andrew Stuart, and Anima Anandkumar. Fourier neural operator for parametric partial differential equations. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=c8P9NQVtmnO.

Zongyi Li, Nikola Kovachki, Chris Choy, Boyi Li, Jean Kossaifi, Shourya Otta, Mohammad Amin Nabian, Maximilian Stadler, Christian Hundt, Kamyar Azizzadenesheli, and Animashree Anandkumar. Geometry-informed neural operator for large-scale 3d pdes. In A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine (eds.), *Advances in Neural Information Processing Systems*, volume 36, pp. 35836–35854. Curran Associates, Inc., 2023. doi: 10.52202/075280-1556. URL https://proceedings.neurips.cc/paper_files/paper/2023/file/70518ea42831f02afc3a2828993935ad-Paper-Conference.pdf.

Nathan Lichtlé, Alexi Canesse, Zhe Fu, HOSSEIN MATIN, Maria Delle Monache, and Alexandre M Bayen. (u)nfv: (un)supervised neural finite volume methods for solving hyperbolic pdes. In C. Vondrick, B. Hariharan, C. Raffel, L. Pinto, D. Yang, and A. Faust (eds.), *International Conference on Learning Representations*, volume 2026, pp. 77393–77415, 2026. URL https://proceedings.iclr.cc/paper_files/paper/2026/file/7d411ed79f762a0e99337cf4ce38938a-Paper-Conference.pdf.

Phillip Lippe, Bastiaan S. Veeling, Paris Perdikaris, Richard E. Turner, and Johannes Brandstetter. PDE-Refiner: Achieving accurate long rollouts with neural PDE solvers. In *Advances in Neural Information Processing Systems*, volume 36, 2023. URL https://papers.nips.cc/paper_files/paper/2023/hash/d529b943af3dba734f8a7d49efcb6d09-Abstract-Conference.html.

Ning Liu, Yiming Fan, Xianyi Zeng, Milan Klöwer, Lu Zhang, and Yue Yu. Harnessing the power of neural operators with automatically encoded conservation laws. In Ruslan Salakhutdinov, Zico Kolter, Katherine Heller, Adrian Weller, Nuria Oliver, Jonathan Scarlett, and Felix Berkenkamp (eds.), *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 30965–30997. PMLR, 21–27 Jul 2024. URL https://proceedings.mlr.press/v235/liu24p.html.

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017.

Lu Lu, Pengzhan Jin, Guofei Pang, Zhongqiang Zhang, and George Em Karniadakis. Learning nonlinear operators via DeepONet based on the universal approximation theorem of operators. *Nature Machine Intelligence*, 3:218–229, 2021.

Andreas K. Maier, Christopher Syben, Bernhard Stimpel, Tobias Würfl, Mathis Hoffmann, Frank Schebesch, Weilin Fu, Leonid Mill, Lasse Kling, and Silke Christiansen. Learning with known operators reduces maximum error bounds. *Nature Machine Intelligence*, 1:373–380, 2019. doi: 10.1038/s42256-019-0077-5.

Guido Montúfar, Razvan Pascanu, Kyunghyun Cho, and Yoshua Bengio. On the number of linear regions of deep neural networks. *Advances in neural information processing systems*, 27, 2014.

Tobias Pfaff, Meire Fortunato, Alvaro Sanchez-Gonzalez, and Peter Battaglia. Learning mesh-based simulation with graph networks. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=roNqYL0_XP.

Hendrik Ranocha, Michael Schlottke-Lakemper, Andrew R. Winters, Erik Faulhaber, Jesse Chan, and Gregor J. Gassner. Adaptive numerical simulations with Trixi.jl: A case study of Julia for scientific computing. *Proceedings of the JuliaCon Conferences*, 1(1):77, 2022. doi: 10.21105/jcon.00077.

Bogdan Raonic, Roberto Molinaro, Tim De Ryck, Tobias Rohner, Francesca Bartolucci, Rima Alaifari, Siddhartha Mishra, and Emmanuel De Bézenac. Convolutional neural operators for robust and accurate learning of pdes. *Advances in Neural Information Processing Systems*, 36: 77187–77200, 2023.

Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention*, pp. 234–241. Springer, 2015.

Alvaro Sanchez-Gonzalez, Jonathan Godwin, Tobias Pfaff, Rex Ying, Jure Leskovec, and Peter Battaglia. Learning to simulate complex physics with graph networks. In Hal Daumé III and Aarti

Singh (eds.), *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pp. 8459–8468. PMLR, 13–18 Jul 2020. URL https://proceedings.mlr.press/v119/sanchez-gonzalez20a.html.

Makoto Takamoto, Timothy Praditia, Raphael Leiteritz, Daniel MacKinlay, Francesco Alesiani, Dirk Pflüger, and Mathias Niepert. Pdebench: An extensive benchmark for scientific machine learning. In S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh (eds.), *Advances in Neural Information Processing Systems*, volume 35, pp. 1596–1611. Curran Associates, Inc., 2022. doi: 10.52202/068431-0117. URL https://proceedings.neurips.cc/paper_files/paper/2022/file/0a9747136d411fb83f0cf81820d44afb-Paper-Datasets_and_Benchmarks.pdf.

Alasdair Tran, Alexander Mathews, Lexing Xie, and Cheng Soon Ong. Factorized fourier neural operators. *arXiv preprint arXiv:2111.13802*, 2021.

Niklas Wintermeyer, Andrew R Winters, Gregor J Gassner, and David A Kopriva. An entropy stable nodal discontinuous galerkin method for the two dimensional shallow water equations on unstructured curvilinear meshes with discontinuous bathymetry. *Journal of Computational Physics*, 340:200–242, 2017.

Haixu Wu, Huakun Luo, Haowen Wang, Jianmin Wang, and Mingsheng Long. Transolver: A fast transformer solver for PDEs on general geometries. In Ruslan Salakhutdinov, Zico Kolter, Katherine Heller, Adrian Weller, Nuria Oliver, Jonathan Scarlett, and Felix Berkenkamp (eds.), *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 53681–53705. PMLR, 21–27 Jul 2024. URL https://proceedings.mlr.press/v235/wu24r.html.

Xiangxiong Zhang and Chi-Wang Shu. On positivity-preserving high order discontinuous galerkin schemes for compressible euler equations on rectangular meshes. *Journal of Computational Physics*, 229(23):8918–8934, 2010. ISSN 0021-9991. doi: https://doi.org/10.1016/j.jcp.2010.08.016. URL https://www.sciencedirect.com/science/article/pii/S0021999110004535.

# A THEORY: STATEMENTS, ASSUMPTIONS, AND PROOFS

The analysis follows the three questions in Sections 3–4: inherited local balance, representation through known nonlinear computation, and compatibility of discrete invariants. The final result quantifies how much finite calibration can compensate for mismatch.

For $n$ cells and $d$ conservative channels, fix positive volumes $V_i$ and write

$$M = \mathrm{diag}(V_i) \otimes I_d, \qquad \langle x, y \rangle_M = x^\mathsf{T} M y, \qquad \|x\|_M^2 = \sum_i V_i \|x_i\|_2^2. \tag{7}$$

For a fixed input law $\mu$, let $\mathcal{H} = L^2(\mu; \mathbb{R}^{nd}, M)$, with population norm $\|g\|_{\mathcal{H}}^2 = \mathbb{E}_{U\sim\mu}\|g(U)\|_M^2$. Targets and predictors below are measurable, and the stated population identities assume finite second moments. Our deterministic target is an increment $m(U)$. If observations $Y$ are random, take $m(U) = \mathbb{E}[Y \mid U]$; expanding the square and conditioning on $U$ gives

$$\mathbb{E}\|g(U) - Y\|_M^2 = \|g - m\|_{\mathcal{H}}^2 + \mathbb{E}\|Y - m(U)\|_M^2. \tag{8}$$

Indeed, the cross term has conditional expectation zero. Thus approximation errors translate into squared prediction risks by adding a common irreducible term. These volume-weighted norms are the metrics of the results below; the reported macro-RMSE averages channel and trajectory errors differently.

## A.1 EXACT FINITE-VOLUME BALANCE

Choose one unit normal $n_f$ on each internal face. Its outward incidence $s_{if}$ is $+1$ on the cell from which $n_f$ points and $-1$ on the other cell. On an external boundary face choose the outward normal and put $s_{if} = +1$. A periodic pair is treated as one internal face. Let $A_f > 0$ be the face measure and let $\widehat{\Phi}_f \in \mathbb{R}^d$ be one shared normal flux, evaluated once per internal face. All side-specific source corrections are included exactly once in $\widehat{S}_i$.

**Proposition A.1** (Balance on every union of cells)**.** *For arbitrary reconstructed states and shared fluxes, the update*

$$V_i \Delta U_i = -\tau \sum_{f \subset \partial K_i} A_f s_{if} \widehat{\Phi}_f + \tau V_i \widehat{S}_i \tag{9}$$

*satisfies, on every collection of cells $\mathcal{I}$,*

$$\sum_{i\in\mathcal{I}} V_i \Delta U_i = -\tau \sum_{f\in\partial\mathcal{I}} A_f s_{\mathcal{I}f} \widehat{\Phi}_f + \tau \sum_{i\in\mathcal{I}} V_i \widehat{S}_i. \tag{10}$$

*Here $\partial\mathcal{I}$ consists of faces with exactly one incident cell in $\mathcal{I}$, including external faces, and $s_{\mathcal{I}f}$ is that cell's outward sign. In particular, summing over all cells gives Eq. (4). On a closed or periodic mesh, every channel with zero total source is conserved.*

*Proof.* Sum Eq. (9) over $i \in \mathcal{I}$. An internal face with both incident cells in $\mathcal{I}$ appears twice, with the same $A_f\widehat{\Phi}_f$ and opposite signs, and contributes zero. A face with one incident cell in $\mathcal{I}$ appears once and contributes its outward flux. The source terms sum directly. This proves Eq. (10) and its full-domain specialization.

□

For completeness, the exact PDE averages obey the same integral structure. Suppose $u$ is a weak solution of $\partial_t u + \nabla\cdot F(u) = S$ with integrable normal flux traces on these cell faces and with the corresponding space–time balance valid. Set $U_i(t) = V_i^{-1}\int_{K_i} u(x,t)\,dx$ and

$$\overline{\Phi}_f = \frac{1}{\tau A_f}\int_{t_0}^{t_0+\tau}\int_f F(u)\cdot n_f\, dA\, dt, \qquad \overline{S}_i = \frac{1}{\tau V_i}\int_{t_0}^{t_0+\tau}\int_{K_i} S\, dx\, dt. \tag{11}$$

Integrating the weak conservation law over the cell's space–time cylinder (or approximating its indicator by test functions and using the stated traces) gives

$$U_i(t_0+\tau) - U_i(t_0) = -\frac{\tau}{V_i}\sum_f A_f s_{if}\overline{\Phi}_f + \tau\overline{S}_i. \tag{12}$$

Consequently, numerical conservation holds independently of whether the numerical flux approximates these time-averaged physical fluxes well. The construction below makes them equal exactly.

A paired diffusive flux can be included in $\widehat{\Phi}_f$ without changing the proof. For topographic shallow water the bed source has zero mass component, so mass is conserved on the periodic mesh while the total momentum includes the bed contribution. For open boundaries the boundary term is retained. The identity is in exact arithmetic and applies to the stated update: if an executed method adds a post-update correction $c_i$, its balance additionally contains $\sum_i V_i c_i$. It is the shared transport term and the declared source/boundary accounting that establish conservation, not the use of a numerical flux alone.

### A.2 A CONSTRUCTIVE REPRESENTATION ADVANTAGE

We now compare prescribed nonlinear flux computation with a complete capacity-limited direct prediction class. The assumptions concern a real entropy-solution family and a declared input distribution. The comparison does not require a spatial frequency cutoff.

**Lemma A.2** (Sharp polynomial interval certificate)**.** *Let $s \geq 1$ be an integer, $f \in C^s([a_0, a_0+\ell])$, and $f^{(s)} \geq \kappa > 0$. Every polynomial $p$ of degree at most $s-1$ satisfies*

$$\int_{a_0}^{a_0+\ell} |f-p|^2\, da \geq \frac{\kappa^2 \ell^{2s+1}}{C_s^2}, \qquad C_s = s!\binom{2s}{s}\sqrt{2s+1}. \tag{13}$$

*The constant is sharp. A vector-valued version holds if a fixed unit linear functional of $f^{(s)}$ is at least $\kappa$.*

*Proof.* Translate the interval to $[0,\ell]$ and put $w(a) = a^s(\ell-a)^s$. Its derivatives of orders $0,\dots,s-1$ vanish at both endpoints. Repeated integration by parts gives

$$\left|\int_0^\ell (f-p)w^{(s)}\, da\right| = \left|\int_0^\ell f^{(s)} w\, da\right| \geq \kappa\frac{(s!)^2\ell^{2s+1}}{(2s+1)!}.$$

Integrating by parts another $s$ times, using $w^{(2s)} = (-1)^s(2s)!$, gives directly

$$\int_0^\ell (w^{(s)})^2\, da = (-1)^s\int_0^\ell w\, w^{(2s)}\, da = (2s)!\int_0^\ell w\, da = \frac{(s!)^2\ell^{2s+1}}{2s+1}.$$

The same integrations show that $w^{(s)}$ is orthogonal to every lower-degree polynomial. Cauchy–Schwarz now proves (13). For $f = \kappa a^s/s!$, the residual after projection onto degree-$s-1$ polynomials is a multiple of $w^{(s)}$, giving equality. Apply a unit functional for the vector version.

□

Let $a \in I$ with $|I| = L > 0$, and let $z$ denote nuisance parameters. Assume a marginal law $\nu(dz)$ and conditional density $p(a \mid z) \geq \beta/L$ for almost every $(a,z)$, with $0 < \beta \leq 1$; $\mu$ is their pushforward

under $U = X(a, z)$. The embedding need only be affine in $a$ for each fixed $z$; joint affinity is unnecessary. For an integer $P \geq 1$, let $\mathcal{B}_{P,s-1}$ be a class whose *complete increment*, restricted to each such input line, is polynomial of degree at most $s-1$ on at most $P$ intervals covering $I$. Partitions can depend on both the predictor and $z$. Projecting onto a fixed $M$-unit vector, summing (13), and using $\sum_{j=1}^{p} \ell_j^{2s+1} \geq L^{2s+1}/p^{2s}$, $p \leq P$, gives

$$\inf_{b \in \mathcal{B}_{P,s-1}} \|G - b\|_{\mathcal{H}} \geq \frac{\kappa L^s \sqrt{\beta}}{C_s P^s} \tag{14}$$

whenever $a \mapsto G(X(a, z))$ is $C^s$ and $\langle v, \partial_a^s [G(X(a, z))] \rangle_M \geq \kappa$ for one fixed $M$-unit $v$, every $a \in I$, and $\nu$-almost every $z$. The density lower bound contributes $\beta/L$ before integration over $z$; taking the square root yields (14).

**The PDE family and reconstruction.** Use the unit periodic line, uniform cells $K_i = [ih, (i+1)h)$, $h = 1/N$, and $M = hI$. Write $Q_h u = (h^{-1} \int_{K_i} u \, dx)_i$ for cell averaging and $e_i$ for the coordinate vectors. Choose two nonadjacent cells $K_{j_L}, K_{j_R}$ in cyclic order, with at least one full plateau cell between them in each direction. Choose the coordinate origin so that $x_L < x_R$. The two initial jump locations $x_L \in K_{j_L}$, $x_R \in K_{j_R}$ vary in compact interior intervals at distance at least $d_0 > 0$ from cell faces. Let $z = (x_L, x_R)$ and

$$u_0(x; a, z) = c + a \mathbf{1}_{(x_L, x_R)}(x), \qquad a \in I = [a_-, a_+], \quad 0 < a_- < a_+. \tag{15}$$

Assume $f$ is strictly convex on $[c, c + a_+]$, smooth enough for the derivative used below, and set

$$\Lambda = \max_{u \in [c, c+a_+]} |f'(u)|, \qquad 0 < T < d_0/\Lambda. \tag{16}$$

Strict convexity on a nondegenerate state interval ensures $\Lambda > 0$.

**Theorem A.3** (Exact finite-time entropy-wave separation)**.** *For $u_t + f(u)_x = 0$ with periodic initial data (15), there is a fixed, local, linear, constant-reproducing reconstruction of two equal states at each face such that* any consistent numerical flux *gives the exact cell-average increment for every $0 < \tau \leq T$:*

$$m_\tau(U) = Q_h u(\tau) - Q_h u_0 = \tau \, [f(c + a) - f(c)] \, w, \qquad w = \frac{e_{j_R} - e_{j_L}}{h}. \tag{17}$$

*The same fixed reconstruction gives an exact rollout for any subdivision whose cumulative time does not exceed $T$.*

*Suppose $f^{(s)}(c + a)$ has a fixed sign and $|f^{(s)}(c + a)| \geq \kappa_f > 0$ on $I$, for an integer $s \geq 2$. For the conditional law and complete predictor class above,*

$$\inf_{b \in \mathcal{B}_{P,s-1}} \|b - m_\tau\|_{\mathcal{H}} \geq \Gamma_{P,s} := \frac{\tau \sqrt{2/h} \, \kappa_f L^s \sqrt{\beta}}{C_s P^s} > 0, \qquad \inf_{p \in \mathcal{P}} \|p - m_\tau\|_{\mathcal{H}} = 0. \tag{18}$$

*Here $\mathcal{P}$ contains the displayed linear reconstruction and the prescribed flux update. The direct class may additionally be required to conserve total mass; the lower bound still holds. The admissible time $T$ is independent of $P$.*

*Proof.* **1. The true entropy solution.** At $x_L$, the low-to-high jump produces the rarefaction

$$u(x, t) = \begin{cases} c, & (x - x_L)/t \leq f'(c), \\ (f')^{-1}((x - x_L)/t), & f'(c) < (x - x_L)/t < f'(c + a), \\ c + a, & (x - x_L)/t \geq f'(c + a). \end{cases}$$

Strict convexity makes $f'$ strictly increasing. At $x_R$, the high-to-low jump is a shock of speed

$$s(a) = \frac{f(c+a) - f(c)}{a}. \tag{19}$$

Strict convexity gives $s'(a) = [f'(c+a) - s(a)]/a > 0$, so the shock speed is genuinely amplitude-dependent. It satisfies Rankine–Hugoniot. More explicitly, for a convex $C^2$ entropy $\eta$ and entropy flux $q'_\eta = \eta' f'$, with jumps taken from the high state to the low state, its entropy production is

$$[q_\eta] - s(a)[\eta] = \int_c^{c+a} \eta''(v)\big[f(v) - f(c) - s(a)(v-c)\big]\, dv \le 0.$$

The identity follows by integration by parts; the inequality follows because a convex flux lies below its secants. Approximation gives the nonsmooth convex entropy inequalities as well. Together with the continuous rarefaction fan, these are the standard entropy Riemann solutions (Ketcheson et al., 2020). Their wave speeds lie between the endpoint characteristic speeds and hence have absolute value at most $\Lambda$. By (16), each wave remains strictly inside its initial cell, so the two solutions can be joined through constant plateaus with no interaction or face crossing. This explicitly constructs the entropy solution on $[0, T]$; standard scalar entropy uniqueness identifies it as the target.

**2. Linear reconstruction without future information.** At every mesh face, the solution is constant throughout $[0, T]$, equal either to $c$ or to $c + a$. For each face select an adjacent cell that is not one of the two jump cells and has that same plateau value. Such a cell exists because the jump cells are nonadjacent. Copy its current average to both sides of the face. The selector is fixed for this declared family, local, linear, and sends every global constant state to itself. It does not use a future label or a subcell position. The cell averages identify $a$ from a full high plateau and the positions from the two jump-cell averages; since $a > 0$, the parameter embedding is injective and affine in $a$ for fixed $z$.

**3. Exact finite-volume balance.** Consistency gives $\widehat{f}(q, q) = f(q)$. All face values are constant in time, so the numerical flux is the true time-averaged flux at every face. Integrating the conservation law on each cell gives (17): only $K_{j_L}$ and $K_{j_R}$ change, by opposite amounts. All copied plateau cells stay unchanged. Repeating the same argument inductively proves exact subdivision up to cumulative time $T$.

**4. Independent direct-predictor lower bound.** Since $\|w\|_M = \sqrt{2/h}$, choose $v = \pm w / \|w\|_M$ so that

$$\langle v, \partial_a^s m_\tau \rangle_M = \tau \sqrt{2/h}\, |f^{(s)}(c+a)| \ge \tau \sqrt{2/h}\, \kappa_f.$$

Apply (14). Restricting the direct class to conservative predictors can only increase its infimum; moreover, the witness $v$ itself has zero total mass. Thus the obstruction concerns the shape of a conservative increment. No Taylor remainder is used, and no step of the proof depends on $P$ through $T$.

□

**Scope and perturbations.** For the piecewise-affine comparison take $s = 2$ and an amplitude interval on which $f'' \ge \kappa_f > 0$. Such an interval exists for every smooth strictly convex flux: $f'' \ge 0$, and $f''$ cannot vanish identically on a nondegenerate state interval, since this would make $f$ affine there. Continuity therefore provides a compact subinterval on which $f'' \ge \kappa_f > 0$; choose $c + I$ inside it with $a_- > 0$. This proves Proposition 4.1. The selector is fixed for this family; it is not asserted to be the selector learned by our graph processor. The no-face-crossing horizon is typically $O(h)$, and is independent of the direct-class complexity $P$.

A complete ReLU network restricted to an affine input line is piecewise affine (Montúfar et al., 2014). A subsequent polynomial readout changes its degree, and smooth activations or state-dependent normalization require a different capacity analysis. The theorem isolates a representation mechanism for the specified direct class; it does not compare our graph models or rank interface-state and unrestricted shared-flux decoders.

If a perturbed target $\widetilde{m}$ and implemented numerical predictor $p$ satisfy $\|\widetilde{m} - m_\tau\|_{\mathcal{H}} \le \delta$ and $\|p - m_\tau\|_{\mathcal{H}} \le \varepsilon$, triangle inequalities give

$$\|p - \widetilde{m}\|_{\mathcal{H}} \le \varepsilon + \delta, \qquad \inf_{b} \|b - \widetilde{m}\|_{\mathcal{H}} \ge (\Gamma_{P,s} - \delta)_+. \tag{20}$$

Hence $\varepsilon + 2\delta < \Gamma_{P,s}$ preserves a strict advantage. Any use of this perturbation conclusion requires the displayed bounds for the particular target and implementation.

### A.3 Error floor from incompatible discrete invariants

Consider a connected closed graph with $n$ cells, $e$ unique internal faces, and oriented incidence $B \in \mathbb{R}^{n\times e}$. Let $M_* = \operatorname{diag}(V_i^*)$ be the evaluation/target volume matrix and $\widetilde{M} = \operatorname{diag}(\widetilde{V}_i)$ the fixed assembly volume matrix. Use integrated face contributions, so positive areas, shared calibration, and time factors can be included in $q$. For one state channel, every raw shared-flux update has the form

$$r_\theta(U) = \widetilde{M}^{-1} B q_\theta(U). \tag{21}$$

The map $q_\theta$ may be an arbitrary nonlinear measurable function. For a prescribed source/boundary contribution $b(U)$, apply the theorem to the residual target $r(U) = m(U) - b(U)$; $b$ must actually be known and consistently subtracted.

**Theorem A.4** (Exact error floor under an incompatible invariant)**.** *Write $\widetilde{v} = \widetilde{M}\mathbf{1}$. For any target rate or increment $r$,*

$$\inf_{q\in\mathbb{R}^e} \left\| \widetilde{M}^{-1} B q - r \right\|_{M_*}^2 = \frac{|\widetilde{v}^T r|^2}{\widetilde{v}^T M_*^{-1} \widetilde{v}}. \tag{22}$$

*The closest reachable increment is*

$$r_{\mathrm{reach}} = r - M_*^{-1}\widetilde{v}\,\frac{\widetilde{v}^T r}{\widetilde{v}^T M_*^{-1}\widetilde{v}}. \tag{23}$$

*Consequently every nonlinear shared-flux decoder satisfies the corresponding expected lower bound, with the numerator replaced by $\mathbb{E}|\widetilde{v}^T r(U)|^2$. With unconstrained measurable flux functions this expected bound is attained.*

*All target-volume-conservative updates can be represented by the wrong-volume shared assembly if and only if*

$$\widetilde{V}_i = cV_i^* \quad \textit{for every cell, for one } c > 0. \tag{24}$$

*The statement extends channelwise to systems. It is independent of the PDE flux formula.*

*Proof.* Connectivity gives $\ker B^T = \operatorname{span}\{\mathbf{1}\}$: if $B^T x = 0$, the values of $x$ agree at both ends of every edge and hence everywhere. It follows by rank and orthogonality that $\operatorname{range} B = \mathbf{1}^\perp$. Therefore

$$\operatorname{range}(\widetilde{M}^{-1}B) = \ker(\widetilde{v}^T).$$

The $M_*$-normal to this hyperplane is $M_*^{-1}\widetilde{v}$. Its orthogonal projection is (23), and its squared projection distance is (22). Explicitly, let $B^\dagger$ be the Moore–Penrose pseudoinverse and choose $q(U) = B^\dagger \widetilde{M} r_{\mathrm{reach}}(U)$. Since $\widetilde{M} r_{\mathrm{reach}}(U) \in \operatorname{range} B$, this flux attains the projected state. Composing the fixed linear maps with the measurable target proves attainability in the unrestricted measurable class. Integrating proves the expected formula. Both the target and wrong-volume zero-total spaces have codimension one; inclusion of the first in the second is equivalent to equality, hence to proportional normals $\widetilde{v} = cM_*\mathbf{1}$. Positivity gives $c > 0$.

□

The obstruction depends on the target's excitation of the wrong invariant: nonproportional volumes need not give positive error for every input. Positive face-area changes alone can be absorbed into unrestricted face fluxes and do not create this particular invariant error. For example, $V^* = (1, 1)$, $\widetilde{V} = (1, 2)$, and $r = (1, -1)$ give the closest reachable vector $(1.2, -0.6)$ and squared error $1/5$. A common volume rescaling leaves the reachable hyperplane unchanged. This proves Eq. (6) and its attainability claim.

### A.4 Finite calibration: compensable and residual mismatch

An invariant obstruction is global. More general geometric, source, or observation mismatch may instead be partly compensable. The next result specifies what a finite calibration can achieve in a verified parameter neighborhood.

**Theorem A.5** (Exact finite-budget compensation and nonlinear enclosure)**.** *Fix a finite-dimensional Euclidean parameter space, a real output Hilbert space $\mathcal{H}$, and $\rho \geq 0$. Parameter norms are Euclidean; output norms and inner products are those of $\mathcal{H}$. Suppose on $\|w\| \leq \rho$,*

$$g(w) - m = e + Jw + N(w), \qquad e = g(0) - m, \qquad \sup_{\|w\| \leq \rho} \|N(w)\|_{\mathcal{H}} \leq \omega(\rho),$$

*where $J$ is the complete parameter Jacobian or another independently verified affine correction operator. Let $P$ project onto* range $J$*, and define*

$$d(\rho) = \min_{\|w\| \leq \rho} \|e + Jw\|_{\mathcal{H}}.$$

*Then*

$$d(\rho)^2 = \|(I - P)e\|^2 + \min_{\|w\| \leq \rho} \|Pe + Jw\|^2, \tag{25}$$

$$(d(\rho) - \omega(\rho))_+ \leq \inf_{\|w\| \leq \rho} \|g(w) - m\| \leq d(\rho) + \omega(\rho). \tag{26}$$

*If $J = Dg(0)$ and $g$ is $C^2$ on a neighborhood of the closed ball, Taylor's integral formula permits $\omega(\rho) = \frac{1}{2} H_\rho \rho^2$, where $H_\rho = \sup_{\|w\| \leq \rho} \|D^2 g(w)\|_{\mathrm{op}}$ uses the bilinear operator norm into $\mathcal{H}$. For a different linear operator $J$, the valid bound is instead $\omega(\rho) = \rho \|Dg(0) - J\|_{\mathrm{op}} + \frac{1}{2} H_\rho \rho^2$. Nonsmooth models can use any independently valid remainder modulus.*

*Write $J = \sum_{j=1}^{s} \sigma_j u_j v_j^*$ with $\sigma_j > 0$, $a_j = \langle e, u_j \rangle$, $e_\perp = (I - P)e$, and $\rho_{\mathrm{LS}}^2 = \sum_j a_j^2 / \sigma_j^2$. If $\rho \geq \rho_{\mathrm{LS}}$, then $d(\rho) = \|e_\perp\|$ and the minimum-norm minimizer is $w_{\mathrm{LS}} = -\sum_j a_j v_j / \sigma_j$. If $0 < \rho < \rho_{\mathrm{LS}}$, the unique $\lambda > 0$ satisfying*

$$\sum_j \frac{\sigma_j^2 a_j^2}{(\sigma_j^2 + \lambda)^2} = \rho^2 \tag{27}$$

*gives*

$$w_\rho = -\sum_j \frac{\sigma_j a_j}{\sigma_j^2 + \lambda} v_j, \quad d(\rho)^2 = \|e_\perp\|^2 + \sum_j \frac{\lambda^2 a_j^2}{(\sigma_j^2 + \lambda)^2}. \tag{28}$$

*For $\rho = 0$, $d(0) = \|e\|$. Empty sums cover $J = 0$.*

*The exact dual formula is*

$$d(\rho) = \max_{\|v\| \leq 1} \{\langle v, e \rangle - \rho \|J^* v\|\}. \tag{29}$$

*Every chosen $v$ with $\|v\| \le 1$ therefore yields the valid nonlinear lower certificate*

$$\inf_{\|w\|\le\rho} \|g(w) - m\| \ge \big(\langle v, e\rangle - \rho\,\|J^* v\| - \omega(\rho)\big)_+.$$

*Proof.* The components in $\operatorname{range} J$ and its orthogonal complement give (25). For every feasible $w$, reverse triangle inequality bounds its nonlinear error below by $\|e + Jw\| - \omega(\rho)$. The affine minimum is attained on the compact parameter ball; use its minimizer for the upper bound. This proves (26).

For the smooth remainder bounds, Taylor's integral formula gives

$$N(w) = (Dg(0) - J)w + \int_0^1 (1-t) D^2 g(tw)[w, w]\, dt.$$

Taking norms yields the stated modulus, with the first term zero when $J = Dg(0)$.

A component of $w$ in $\ker J$ spends budget without changing the objective. In the $v_j$ coordinates it suffices to minimize $\|e_\perp\|^2 + \sum_j (a_j + \sigma_j w_j)^2$ over $\sum_j w_j^2 \le \rho^2$. The least-squares coordinates give the inactive-constraint case. Otherwise the minimum lies on the sphere. Minimizing the objective plus $\lambda(\sum_j w_j^2 - \rho^2)$ gives the coordinates in (28). The left side of (27) is continuous and strictly decreasing from $\rho_{\mathrm{LS}}^2$ to zero, so there is a unique multiplier. The penalized objective at any feasible point is no greater than its original objective and no smaller than its value at $w_\rho$; at $w_\rho$ the penalty vanishes. This proves global constrained optimality. Substitution computes the residual.

For each unit-ball $v$ and feasible $w$, $\|e + Jw\| \ge \langle v, e + Jw\rangle \ge \langle v, e\rangle - \rho\,\|J^* v\|$. For equality, let $r_* = e + Jw_*$ be an optimal residual. If $r_* = 0$, choose $v = 0$. Otherwise choose $v = r_*/\,\|r_*\|$. Convex first-order optimality says $\langle J^* r_*, w - w_*\rangle \ge 0$ for every feasible $w$. Thus $w_*$ minimizes the linear functional $\langle J^* v, w\rangle$ on the ball, attaining $-\rho\,\|J^* v\|$. Equality follows in (29). Subtract the verified remainder for the last assertion.

□

An SVD on a truncated parameter subspace minimizes over a *smaller* feasible set and gives an upper bound on the full affine optimum. Its leftover residual is not an impossibility certificate for the full network. In contrast, a dual witness is valid if $\|J^* v\|$ is bounded for the *full* Jacobian. Parameter norms depend on parameterization; the chosen metric must be fixed before drawing a comparison.

The mismatch decomposition has a direct modeling interpretation: $(I - P)e$ is orthogonal to every permitted first-order correction, while the remaining error depends on both the available parameter budget and the singular values of the complete readout. Correcting a constant scale can be inexpensive when its mismatch lies in a responsive direction; correcting an excluded invariant is impossible within an assembly that preserves that invariant. This distinction motivates the separate fixed, calibrated, and paired geometry interventions.

# B Datasets, numerical adapters, and reproducibility

## B.1 Equations and reference datasets

The suite separates the high-order solver used to generate reference trajectories from the numerical readout used in the learned model. Four regular datasets store observations on $128 \times 128$ periodic grids. Burgers, KPP, and flat shallow water store conservative cell averages. Topographic shallow water first averages the conservative DG state and then stores height and velocity; the model converts these to conservative variables for each update. Euler instead supplies vertex observations on irregular meshes. Table 5 summarizes the reference contracts.

**Conservative vector viscous Burgers.** For $q = (u, v)^\mathsf{T}$ on $[-1, 1]^2$,

Table 5: Dataset contracts. Counts are trajectories; the frame count includes the initial observation. The Euler step is the configured observation interval. DG degree describes the reference calculation.

| Dataset | Train | Test | Frames | DG degree | Observation time step |
|---|---|---|---|---|---|
| Vector Burgers | 270 | 30 | 80 | 4 | $0.400/79$ |
| KPP | 280 | 20 | 80 | 3 | $1.000/79$ |
| Flat shallow water | 280 | 20 | 80 | 3 | $0.250/79$ |
| Topographic shallow water | 270 | 30 | 51 | 3 | $0.010$ |
| Euler | 300 | 20 | 80 | 3 | $0.025$ |

$$\partial_t q + \nabla \cdot \left(\tfrac{1}{2} q \otimes q\right) = \nu \Delta q, \qquad \nu = 5 \times 10^{-4}. \tag{30}$$

This is the conservative vector equation recorded by the generator. The random initial family combines background velocity, smooth elliptical fronts, Fourier perturbations, vortices, and shear. The reference uses degree-four DG, local Lax–Friedrichs surface fluxes, central split-form volume terms with Hennemann–Gassner shock capturing, and BR1 viscous discretization (Bassi & Rebay, 1997). Hyperbolic and parabolic CFL parameters are 0.35 and 0.1. Eighty frames cover $[0, 0.4]$. The stored graph has 16 384 cells and 32 768 unique periodic faces, with face measures, normals, and cell volumes.

**KPP conservation law.** The scalar nonconvex problem is (Kurganov et al., 2007)

$$\partial_t u + \partial_x \sin u + \partial_y \cos u = 0 \quad \text{on } [-2, 2]^2. \tag{31}$$

The reference uses degree-three DG, an entropy-conservative volume flux, local Lax–Friedrichs surface dissipation, and Hennemann–Gassner shock capturing at CFL 0.8. Eighty saved frames cover $[0, 1]$. The varying initial states generate interacting fronts under a nonconvex flux, complementing the coupled-state tasks.

**Flat shallow water.** Writing $m = (hu, hv)^\mathsf{T}$,

$$\partial_t h + \nabla \cdot m = 0, \qquad \partial_t m + \nabla \cdot \left(\frac{m \otimes m}{h} + \frac{1}{2} g h^2 I\right) = 0, \qquad g = 9.81. \tag{32}$$

The periodic domain is $[-2, 2]^2$. Fully wet randomized elliptical dam-break and weak-blast states generate moving fronts. The degree-three reference uses HLL/Fjordholm surface treatment, Wintermeyer volume terms, Hennemann–Gassner shock capturing, SSPRK43, CFL 0.6, and a positivity treatment for height. Eighty frames cover $[0, 0.25]$.

**Topographic shallow water.** On $[-1, 1]^2$ the momentum equation becomes

$$\partial_t m + \nabla \cdot \left(\frac{m \otimes m}{h} + \frac{1}{2} g h^2 I\right) = -g h \nabla b, \qquad g = 1, \tag{33}$$

with static bathymetry $b$. Initial states combine smooth random bathymetry, free-surface perturbations, vortical/compressive velocities, and background flow while retaining wet cells. The degree-three reference uses a Chen–Noelle hydrostatic HLL surface treatment, Wintermeyer volume terms (Wintermeyer et al., 2017), shallow-water shock capturing, SSPRK43, CFL 0.5, and positive-height limiting. It saves 51 frames on $[0, 0.5]$. The archive also contains other DG degrees; the reported reference contract uses degree three. Bathymetry is stored once per trajectory.

**Compressible Euler.** With conservative state $U = (\rho, \rho u, \rho v, E)^\mathsf{T}$,

$$\partial_t U + \partial_x F_x(U) + \partial_y F_y(U) = 0, \qquad p = (\gamma - 1)\left(E - \tfrac{1}{2}\rho(u^2 + v^2)\right), \quad \gamma = 1.4. \tag{34}$$

A bump-channel configuration uses irregular P4est meshes (Burstedde et al., 2011), degree-three shock-capturing DGSEM, local Lax–Friedrichs surface fluxes, Ranocha entropy-conservative volume terms, SSPRK43, and Zhang–Shu positivity limiting (Hennemann et al., 2021; Zhang & Shu, 2010). The files contain density, velocity, pressure, Mach information, node types, coordinates, and edges. The first training mesh has 19 531 vertices. These are vertex samples of the reference field, not finite-volume averages. Boundary states are prescribed during the reported rollout protocol. Dual-cell reconstruction and point-estimated geometry are therefore separate model configurations, as are fixed and calibrated coefficients.

**Reference software and metadata.** Burgers and KPP use Julia 1.10.11 and Trixi 0.16.13; the flat-water generator additionally records TrixiShallowWater 0.2.9. The topographic archive records Julia 1.12.6, Trixi 0.17.2, TrixiShallowWater 0.3.1, and OrdinaryDiffEqSSPRK 2.2.2.

### B.2 Equation adapters

Every adapter consumes reconstructed equation-specific interface states, performs any required conversion to conservative variables, and returns a unique oriented transport flux. Reversing a face reverses the transported exchange. The generic local Lax–Friedrichs (Rusanov) form is

$$\widehat{\Phi}(U_L, U_R; n) = \frac{F_n(U_L) + F_n(U_R)}{2} - \frac{\alpha}{2}(U_R - U_L). \tag{35}$$

For Burgers $F_n(q) = \frac{1}{2}(q \cdot n)q$, and the implemented speed estimate is $\alpha = \max(|q_L \cdot n|, |q_R \cdot n|)$. The centered viscous face contribution is $-\nu(q_j - q_i)/d_f$, evaluated from current cell states and added to the transport flux. For KPP, $F_n(u) = n_x \sin u + n_y \cos u$ and $\alpha = \|n\|_2 = 1$.

For shallow water, $v_n = v \cdot n$ and $c = \sqrt{gh}$. The HLL speeds are $s_L = \min(0, v_{n,L} - c_L, v_{n,R} - c_R)$ and $s_R = \max(0, v_{n,L} + c_L, v_{n,R} + c_R)$. For $s_L < 0 < s_R$,

$$\widehat{\Phi}_{\mathrm{HLL}} = \frac{s_R F_n(U_L) - s_L F_n(U_R) + s_L s_R (U_R - U_L)}{s_R - s_L}, \tag{36}$$

with the corresponding one-sided flux if all waves have the same sign. The topographic adapter uses hydrostatic reconstruction (Chen & Noelle, 2017): for $\eta_K = h_K + b_K$,

$$b_* = \min(\max(b_L, b_R), \min(\eta_L, \eta_R)), \qquad h_K^* = \min(\eta_K - b_*, h_K). \tag{37}$$

A small positive floor is applied in the numerical implementation. The shared HLL flux uses the reconstructed depths and velocities. Each side receives its own momentum correction $\frac{1}{2}g(h_K^2 - (h_K^*)^2)n$ with the appropriate orientation. Its mass component is zero; the volume-weighted bed force is a source, not an unpaired transport flux.

For Euler, the Rusanov speed estimate combines the maximum normal velocity magnitude with the maximum sound speed $\sqrt{\gamma p/\rho}$. Primitive interface states are converted to conservative density, momentum, and energy before flux evaluation. The implementation uses softplus on interface height, density, and pressure channels. This is an interface parameterization; exact conservation follows from assembly independently of this positivity map.

For topographic shallow water, conversion back to stored height/velocity uses a small height floor to make division well-defined. The audit records both the conservative update and any resulting correction to the stored height. The full-trajectory audit separates input constraints, the raw update, and output conversion, and measures the resulting balance on the recorded states; these fixed-geometry rollouts retain their closed invariants up to the reported floating-point discrepancies.

Table 6: Numerical readouts within a shared interface contract. Reference-generation schemes are specified separately in Appendix B.1.

| Equation | Shared numerical flux | Additional numerical computation |
|---|---|---|
| Vector Burgers | local Lax–Friedrichs | centered viscosity |
| KPP | local Lax–Friedrichs | $F(u) = (\sin u, \cos u)$ |
| Flat shallow water | HLL | conservative momentum update |
| Topographic shallow water | hydrostatic HLL | bed-source correction |
| Compressible Euler | Rusanov | primitive/conservative conversion |

### B.3 Model and optimization

Nodes encode states and static fields; edges encode displacement, normal, and geometric measures. Physical edges define the exchange; additional irregular-mesh neighbors supply context. The neural-operator processor uses state- and geometry-dependent kernels:

$$\widetilde{z}_i^{\ell+1} = W_\ell z_i^\ell + \frac{1}{|\mathcal{N}_i|} \sum_{j \in \mathcal{N}_i} K_\ell(\xi_{ij}^\ell) \odot H_\ell z_j^\ell, \tag{38}$$

followed by layer normalization and a pointwise activation (Ba et al., 2016). The kernel input $\xi_{ij}^\ell$ includes both endpoint features, their difference, and encoded static geometry. A shared edge-message module and component-wise decoder produce one state per directed side,

$$(\widehat{q_f^-}, \widehat{q_f^+}) = D_\theta(z_i^L, z_j^L, e_{ij}). \tag{39}$$

Flat shallow water uses conservative interface coordinates; topographic shallow water and Euler use primitive coordinates. Softplus maps parameterize positive interface channels such as height, density, and pressure. A message-passing processor can replace these kernels while retaining the decoder, equation adapter, and balance update (Appendix D.7).

The default network uses node latent width 120, node encoding width 32, edge encoding width 16, and ten local neural-operator blocks. The regular fixed-geometry models contain approximately 1.196–1.206 million parameters, with small differences due to their state channels. They decode effective left/right interface states directly rather than restricting reconstruction to cell traces plus a small correction. The training target is the stored state, with objective $\mathcal{L}(\theta) = \mathrm{MSE}(\mathcal{N}(\widehat{q}_\theta), \mathcal{N}(q_{\mathrm{obs}}))$, where $\mathcal{N}$ is the training-data normalizer, and MSE is evaluated over supervised nodes. Both one-step and autoregressive training supervise states through the numerical update, without intermediate labels or a conservation penalty. Equation parameters, geometry, static fields, and boundary data are prescribed inputs.

Regular training uses Adam for 20 one-step epochs at learning rate $10^{-4}$ followed by four three-step autoregressive epochs at $10^{-5}$, with gradient clipping at norm one (Kingma & Ba, 2014). Training-noise amplitude follows the task configuration with a three-epoch warmup. The main repeated runs use three independently initialized training seeds. The stored configurations specify state normalizers and task-specific noise scales. The Burgers free-calibration mechanism control uses its recorded TF32 inference path; the fixed-geometry eager benchmark uses IEEE FP32. All other displayed regular mechanism configurations follow their saved precision settings in the result manifest.

Let $T_k$ be the scored time indices of trajectory $k$, $\Omega_k$ its spatial observations, and $C$ the number of state channels. The reported macro error is

$$\mathrm{macroRMSE} = \frac{1}{KC} \sum_{k=1}^{K} \sum_{c=1}^{C} \sqrt{\frac{1}{|T_k||\Omega_k|} \sum_{t \in T_k} \sum_{i \in \Omega_k} (\widehat{q}_{k,t,i,c} - q_{k,t,i,c})^2}. \tag{40}$$

A single-state model scores 79 predictions, or 50 for topography; a five-state model scores its remaining 75 or 46 frames. This metric uses physical state units. Integral diagnostics instead use conservative variables and physical volume weights.

### B.4 Neural baseline configurations

The grid baselines are trained separately for each equation with five input frames, 50 one-step epochs, and 10 three-step autoregressive epochs. Their objective averages relative $L_2$ errors across state channels. Batches contain 32, 128, 96, and 128 examples for Burgers, KPP, flat water, and topographic water, respectively. FNO uses 13 retained modes, width 48, and four Fourier layers (approximately 6.247–6.248 million parameters). U-Net uses base width 29 (6.377–6.380 million parameters) (Ronneberger et al., 2015). DPOT uses patch size eight, 32 modes, width 512, four layers with four blocks, MLP ratio one, and output embedding width 32 (6.193–6.209 million parameters). These DPOT runs learn each equation from its own training trajectories. The recorded DPOT optimizer is Adam with $\beta = (0.9, 0.9)$ and weight decay $10^{-6}$; its OneCycle schedule peaks at $10^{-3}$ after a ten-epoch warmup, with BF16 training.

**FLUXGNN task adaptation.** We adapt FLUXGNN's conservative flux-passing construction (Horie & Mitsume, 2024) from its published convection–diffusion and Navier–Stokes mixture-flow tasks to our four regular-grid equations. It retains a bias-free linear state encoder, a coupled Moore–Penrose pseudoinverse decoder, one shared vector flux per finite-volume face, and equal-and-opposite extensive transfers assembled with face normals, areas, and cell volumes.

The task adapters specify state channels, periodic face connectivity, geometric inputs, and the saved observation interval. For topographic shallow water, stored primitive variables are converted to conservative variables before the update, bathymetry is supplied as a static field, and the known bathymetric momentum source is added explicitly. All three training seeds use the same implementation and task configurations; evaluation starts from one initial state and covers the complete test horizon.

Euler baselines use one input state, 20 one-step epochs and 4 three-step autoregressive epochs, BF16 training, AdamW at $10^{-3}$ with $\beta = (0.9, 0.999)$, and gradient clipping at norm one (Loshchilov & Hutter, 2017). The loss is normalized next-state MSE on normal nodes; the prescribed boundary sequence is shared across methods. GNOT uses width 128, three layers, four heads, two experts, and expansion ratio four (2.928 million parameters), with weight decay $5 \times 10^{-5}$. Its native PyTorch implementation uses masked linear attention. Transolver uses width 256, eight layers and eight heads, 32 slices, and MLP ratio two (3.863 million parameters), with weight decay $10^{-5}$. It retains the irregular-mesh Physics-Attention processor. Both Euler optimizers use the recorded OneCycle schedules; GNOT uses a warmup fraction of 0.2 and initial/final division factors of $10^4$.

### B.5 Graph implementation and deployment

The computation is local on a fixed mesh. Geometry, edge orderings, and physical pairing maps are cached once rather than rebuilt at every predicted step. On the regular grids, the fixed number of neighbors permits structured reductions in place of general scatter operations. Linear endpoint and difference features are folded algebraically: $W_i z_i + W_j z_j + W_\Delta(z_j - z_i)$ equals $(W_i - W_\Delta)z_i + (W_j + W_\Delta)z_j$. The optimized path also removes repeated graph copies and preallocates reused geometry tensors. These operations change execution and storage, while preserving the learned graph computation and paired update.

All numerical cost comparisons use batch-one, resident tensors on one NVIDIA GeForce RTX 5090, IEEE FP32, and eager inference. TF32, autocast, compilation, and cuDNN autotuning are disabled. Deterministic algorithms are enabled, with `CUBLAS_WORKSPACE_CONFIG=:4096:8` set before CUDA initialization. These settings apply to the neural operator and every timed classical scheme. The manuscript source contains the per-run accuracy and timing tables; raw timing rounds and execution records belong to the separate experimental archive.

### B.6 Traditional numerical comparisons

FV1 uses the equation-specific flux and source adapter with adjacent cell states, forward Euler, and CFL-limited substeps landing on each saved observation time. FV2-MC adds MUSCL reconstruction with the monotonized-central limiter and second-order SSP/Heun time integration. FV2-MM1 uses the $\theta = 1$ minmod reconstruction with the corresponding flux family. DG2 uses polynomial degree two on a $256^2$ internal mesh; conservative $2 \times 2$ restriction gives the $128^2$ observation state. Its SSPRK43 integration and source/flux choices follow the reference equation contract.

**Common GPU timing boundary.** FV-GNO, FV1, FV2-MC, and FV2-MM1 run the same saved-time task on the observation grid. Each task uses three common test trajectories: the first, lower-middle, and last in the recorded order. A complete rollout warmup precedes three interleaved measurement rounds, yielding nine timings per trained model or solver. A host timer is synchronized with CUDA before and after each rollout. Timings include all state updates, classical CFL substeps, safety checks, and stage retries; they exclude file I/O, device transfer, loading weights, constructing static graphs, and scoring. This measures a resident predictor or solver over the full saved horizon, without training or reference generation.

**Accuracy and aggregation.** Each timed trajectory is checked against its archived full-test per-variable RMSE outside the timed region. All 162 scalar checks pass the relative tolerance $0.003$ plus absolute tolerance $2 \times 10^{-6}$; the maximum observed relative difference is $0.082\%$. The table uses the full-test score and GPU cost of the same trained model. The timing subset contains Burgers, flat water, KPP, and topography (Table 9). Each cost is the median of nine rollouts; method points average these medians. Horizontal intervals average the within-run p10 and p90 values, including trajectory and repeated-measurement variation.

DG2's available Trixi implementation runs on CPU. We retain its accuracy in Table 3, leave GPU time unavailable, and exclude it from the GPU cost plot and frontier.

# C Full prediction results and numerical comparisons

The tables extend the prediction and solver comparisons in Sections 5.2–5.3, from aggregate error to training-run variation, computational cost, and accuracy near fronts.

## C.1 Repeated training and state-wise prediction

The regular prediction benchmark uses fixed finite-volume geometry. Table 7 reports the prescribed-adapter and external-baseline training runs; Table 14 summarizes the prescribed/shared-flux comparison with sample SDs. Table 8 resolves the main models by state channel.

Table 7: Complete regular-grid training-run results. Dashes denote explosive FLUXGNN rollouts on topographic SWE for all three seeds; their raw errors remain archived. The input/score contracts are those of Table 1.

| Method | Task | 123 | 456 | 789 | Mean | SD |
|---|---|---|---|---|---|---|
| FV-GNO | Burgers | $1.178 \times 10^{-2}$ | $1.282 \times 10^{-2}$ | $1.224 \times 10^{-2}$ | $1.228 \times 10^{-2}$ | $5.219 \times 10^{-4}$ |
| FV-GNO | KPP | $3.646 \times 10^{-2}$ | $3.594 \times 10^{-2}$ | $3.384 \times 10^{-2}$ | $3.541 \times 10^{-2}$ | $1.383 \times 10^{-3}$ |
| FV-GNO | Flat SWE | $1.166 \times 10^{-2}$ | $1.152 \times 10^{-2}$ | $1.185 \times 10^{-2}$ | $1.168 \times 10^{-2}$ | $1.650 \times 10^{-4}$ |
| FV-GNO | SWE + topo. | $4.097 \times 10^{-3}$ | $3.788 \times 10^{-3}$ | $5.323 \times 10^{-3}$ | $4.402 \times 10^{-3}$ | $8.119 \times 10^{-4}$ |
| FLUXGNN | Burgers | $3.926 \times 10^{-2}$ | $3.912 \times 10^{-2}$ | $4.166 \times 10^{-2}$ | $4.001 \times 10^{-2}$ | $1.432 \times 10^{-3}$ |
| FLUXGNN | KPP | $1.915 \times 10^{-1}$ | $1.895 \times 10^{-1}$ | $1.888 \times 10^{-1}$ | $1.900 \times 10^{-1}$ | $1.418 \times 10^{-3}$ |
| FLUXGNN | Flat SWE | $2.880 \times 10^{-2}$ | $2.932 \times 10^{-2}$ | $2.935 \times 10^{-2}$ | $2.916 \times 10^{-2}$ | $3.087 \times 10^{-4}$ |
| FLUXGNN | SWE + topo. | - | - | - | - | - |
| DPOT | Burgers | $3.387 \times 10^{-2}$ | $3.238 \times 10^{-2}$ | $3.336 \times 10^{-2}$ | $3.320 \times 10^{-2}$ | $7.585 \times 10^{-4}$ |
| DPOT | KPP | $1.351 \times 10^{-1}$ | $1.652 \times 10^{-1}$ | $1.332 \times 10^{-1}$ | $1.445 \times 10^{-1}$ | $1.797 \times 10^{-2}$ |
| DPOT | Flat SWE | $1.777 \times 10^{-2}$ | $2.077 \times 10^{-2}$ | $1.799 \times 10^{-2}$ | $1.884 \times 10^{-2}$ | $1.673 \times 10^{-3}$ |
| DPOT | SWE + topo. | $2.362 \times 10^{-2}$ | $2.259 \times 10^{-2}$ | $2.318 \times 10^{-2}$ | $2.313 \times 10^{-2}$ | $5.148 \times 10^{-4}$ |
| FNO | Burgers | $3.664 \times 10^{-1}$ | $1.973 \times 10^{-1}$ | $8.739 \times 10^{-1}$ | $4.792 \times 10^{-1}$ | $3.522 \times 10^{-1}$ |
| FNO | KPP | $3.350 \times 10^{-1}$ | $3.611 \times 10^{-1}$ | $3.232 \times 10^{-1}$ | $3.398 \times 10^{-1}$ | $1.941 \times 10^{-2}$ |
| FNO | Flat SWE | $4.436 \times 10^{-2}$ | $3.897 \times 10^{-2}$ | $4.062 \times 10^{-2}$ | $4.132 \times 10^{-2}$ | $2.762 \times 10^{-3}$ |
| FNO | SWE + topo. | $2.759 \times 10^{-2}$ | $2.740 \times 10^{-2}$ | $2.859 \times 10^{-2}$ | $2.786 \times 10^{-2}$ | $6.396 \times 10^{-4}$ |
| U-Net | Burgers | $7.311 \times 10^{-2}$ | $4.741 \times 10^{-2}$ | $6.619 \times 10^{-2}$ | $6.224 \times 10^{-2}$ | $1.330 \times 10^{-2}$ |
| U-Net | KPP | $1.719 \times 10^{-1}$ | $1.282 \times 10^{-1}$ | $1.650 \times 10^{-1}$ | $1.550 \times 10^{-1}$ | $2.349 \times 10^{-2}$ |
| U-Net | Flat SWE | $7.685 \times 10^{-2}$ | $6.835 \times 10^{-2}$ | $9.495 \times 10^{-2}$ | $8.005 \times 10^{-2}$ | $1.359 \times 10^{-2}$ |
| U-Net | SWE + topo. | $6.067 \times 10^{-2}$ | $6.026 \times 10^{-2}$ | $4.532 \times 10^{-2}$ | $5.542 \times 10^{-2}$ | $8.750 \times 10^{-3}$ |

Table 8: State-wise RMSE for the main models: fixed finite-volume geometry on regular grids and the calibrated Euler extension. Entries list three training runs, their arithmetic mean, and sample standard deviation. SDs below the displayed resolution are shown as $< 0.001$.

| Task | State | 123 | 456 | 789 | Mean | SD |
|---|---|---|---|---|---|---|
| Burgers | u | 0.014 | 0.015 | 0.014 | 0.014 | 0.001 |
| Burgers | v | 0.010 | 0.011 | 0.010 | 0.010 | $< 0.001$ |
| KPP | u | 0.036 | 0.036 | 0.034 | 0.035 | 0.001 |
| Flat SWE | h | 0.006 | 0.006 | 0.006 | 0.006 | $< 0.001$ |
| Flat SWE | hu | 0.015 | 0.014 | 0.015 | 0.015 | $< 0.001$ |
| Flat SWE | hv | 0.015 | 0.014 | 0.015 | 0.015 | $< 0.001$ |
| SWE + topo. | h | 0.004 | 0.004 | 0.006 | 0.005 | 0.001 |
| SWE + topo. | u | 0.004 | 0.004 | 0.005 | 0.004 | 0.001 |
| SWE + topo. | v | 0.004 | 0.004 | 0.005 | 0.004 | 0.001 |
| Euler (calibrated) | $\rho$ | 0.105 | 0.136 | 0.115 | 0.119 | 0.016 |
| Euler (calibrated) | u | 0.047 | 0.062 | 0.053 | 0.054 | 0.007 |
| Euler (calibrated) | v | 0.054 | 0.059 | 0.053 | 0.055 | 0.003 |
| Euler (calibrated) | p | 0.197 | 0.262 | 0.221 | 0.226 | 0.033 |

### C.2 Matched numerical accuracy and rollout cost

Table 9 lists solvers under Appendix B.6. All datasets use all three seeds. The main accuracy–cost table averages the corresponding rows.

Table 9: Matched GPU timing by method or solver. All entries use the same RTX 5090, FP32 IEEE, eager execution, and batch size one. For our method, RMSE and timing statistics are averaged over seeds; the timing statistics for each seed use nine complete rollouts: three common trajectories, each repeated three times after warmup, with interleaved method order. FV baselines are reported once per solver. RMSE is the full-test score of the corresponding trained model or solver, verified on the timed trajectories.

| Task | Method | RMSE | Median (s) | p10 (s) | p90 (s) |
|---|---|---|---|---|---|
| Burgers | FV-GNO | 0.012 | 0.675 | 0.665 | 0.693 |
| Burgers | FV1 | 0.043 | 0.264 | 0.226 | 0.294 |
| Burgers | FV2–MC | 0.016 | 1.712 | 1.445 | 1.936 |
| Burgers | FV2–MM1 | 0.019 | 1.473 | 1.235 | 1.527 |
| KPP | FV-GNO | 0.035 | 0.661 | 0.647 | 0.671 |
| KPP | FV1 | 0.445 | 0.232 | 0.219 | 0.252 |
| KPP | FV2–MC | 0.489 | 1.372 | 1.348 | 1.397 |
| KPP | FV2–MM1 | 0.187 | 2.147 | 2.089 | 2.223 |
| Flat SWE | FV-GNO | 0.012 | 0.741 | 0.716 | 0.764 |
| Flat SWE | FV1 | 0.044 | 0.425 | 0.411 | 0.515 |
| Flat SWE | FV2–MC | 0.013 | 3.074 | 2.952 | 3.280 |
| Flat SWE | FV2–MM1 | 0.021 | 2.709 | 2.630 | 2.947 |
| SWE + topo. | FV-GNO | 0.004 | 0.497 | 0.490 | 0.508 |
| SWE + topo. | FV1 | 0.033 | 0.838 | 0.799 | 0.862 |
| SWE + topo. | FV2–MC | 0.006 | 4.811 | 4.588 | 5.062 |
| SWE + topo. | FV2–MM1 | 0.010 | 4.293 | 4.171 | 4.394 |

### C.3 Capturing discontinuous regions

Reference relative face jumps define the discontinuity regions before inspecting the prediction error. The reported region errors therefore test whether the predictor captures the high-variation part of the field rather than only its smoother background. The regular results use fixed finite-volume predictions; the Euler analysis uses the recorded calibrated model. Figure 5 shows illustrative centerline profiles alongside the aggregate region diagnostics.

Table 10: Discontinuity-region diagnostics of fixed finite-volume models, averaged over available training seeds. Regions are the top 10% of reference cell scores, each the largest incident relative face jump. The last column is reference-front recall within one grid cell.

| Task | Region RMSE | Jump rel. $L_2$ | Mask IoU | Recall |
|---|---|---|---|---|
| Burgers | 0.028 | 0.194 | 0.951 | 0.998 |
| KPP | 0.100 | 0.079 | 0.901 | 0.989 |
| Flat SWE | 0.029 | 0.261 | 0.761 | 0.920 |
| SWE + topo. | 0.012 | 0.166 | 0.922 | 0.991 |

Table 11: Euler error in reference-selected top-10% pressure-jump regions, over 20 trajectories. The interval is the recorded trajectory-bootstrap 95% interval of region macro-RMSE.

| Model | Region RMSE | CI low | CI high |
|---|---|---|---|
| FV-GNO (calibrated) | 0.293 | 0.263 | 0.323 |
| GNOT | 0.394 | 0.314 | 0.482 |
| Transolver | 0.906 | 0.837 | 0.977 |

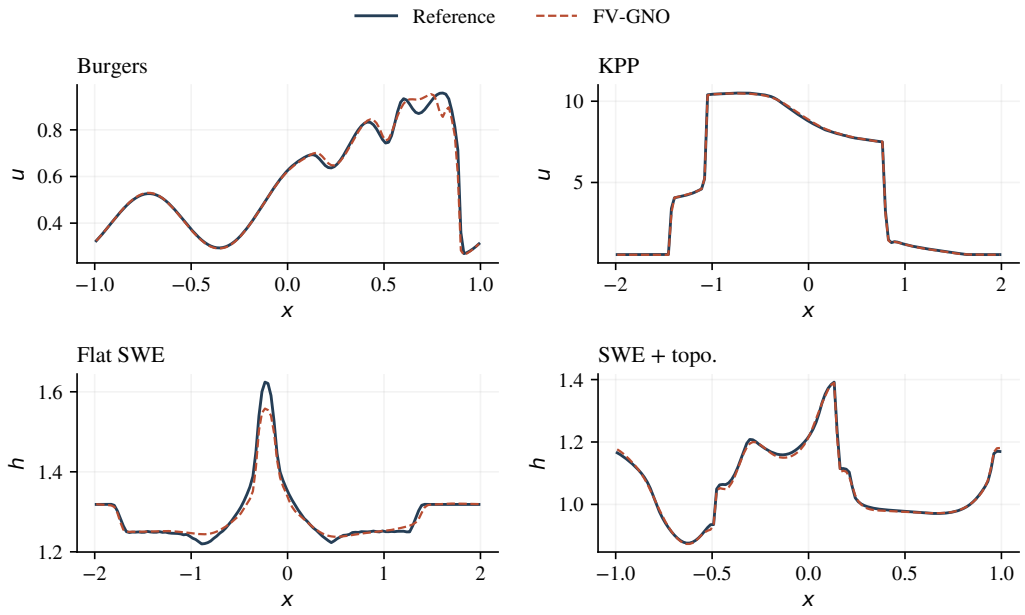


Figure 5: Final-time centerline profiles in saved regular-grid trajectories. High-order reference and fixed finite-volume neural predictions are shown in physical state units. Profiles are illustrative; the accompanying region-error table aggregates the specified evaluation cases.

## D Ablation studies of the finite-volume computation

The ablations follow the computation in Eq. (2): a graph processor supplies an interface approximation, an equation adapter evaluates local interactions, and geometry with paired assembly converts those interactions into a state update. We intervene at these distinct stages to determine what is learned and what the numerical structure contributes. Table 12 summarizes the regular-grid controls. The following subsections distinguish reference-state one-step tests, autonomous prediction, and long-horizon equilibrium retention.

**Interventions and retained computation.** The node controls replace the face representation and local update by direct nodal increments. The cell-trace control replaces only the interface approximation in a fixed numerical update. The geometric controls alter the update coefficients

or remove their explicit $A/V$ factor. Finally, we develop a shared-flux ablation within the same finite-volume framework: its network predicts the interaction rather than its input states. The graph-processor replacement changes the learned representation module while retaining the interface and numerical modules. This organization identifies the operation changed in each experiment, rather than interpreting every control as a test of conservation alone.

Table 12: Component interventions in the regular-grid update: macro-RMSE ($\times 10^{-3}$). These are individual-run controls; the interface-state, shared-flux, node, and projected-node rows use seed 123. Other rows are the corresponding recorded mechanism runs. Repeated node and flux comparisons are reported separately in Tables 13 and 14. A dash indicates an unavailable or nonfinite control.

| Update | Burgers | KPP | Flat SWE | SWE + topo. |
|---|---|---|---|---|
| FV-GNO (prescribed interaction) | 11.782 | 36.456 | 11.663 | 4.097 |
| Shared-flux ablation | 11.728 | 37.084 | 11.436 | 3.660 |
| Free directed calibration | 14.753 | 39.989 | 16.106 | 6.550 |
| No explicit $A/V$ | 12.218 | 34.792 | 11.722 | 5.500 |
| Cell traces + calibration | 40.382 | 123.583 | 21.746 | - |
| Node black box | 20.566 | 46.705 | 17.165 | - |
| Projected node black box | 13.965 | 72.822 | 17.277 | - |

### D.1 Local exchange versus globally projected node prediction

The black-box model predicts a nodal increment. Its projected counterpart applies the volume-weighted global conservation projection inside training, allowing the network to optimize with the correction already present. These controls share the graph-processor family with FV-GNO. Table 13 reports three independently trained models for each configuration, and Fig. 4 places the results alongside the internal shared-flux control.

FV-GNO has lower mean error than both node controls on Burgers, KPP, and flat shallow water. Projection improves the nodal mean but does not recover the accuracy of local finite-volume exchange. Thus, a correct domain total and a structured local computation are different inductive biases: the former restricts the sum of predicted increments, whereas the latter constructs the increments from interactions shared by neighboring cells.

Table 13: Global projection inside training versus fixed finite-volume assembly. Macro-RMSE by training run, followed by mean and sample SD. An SD below the displayed resolution is shown as $< 0.001$.

| Task | Update | 123 | 456 | 789 | Mean | SD |
|---|---|---|---|---|---|---|
| Burgers | Node black box | 0.021 | 0.027 | 0.025 | 0.024 | 0.003 |
| Burgers | Projected node black box | 0.014 | 0.015 | 0.015 | 0.015 | 0.001 |
| Burgers | FV-GNO | 0.012 | 0.013 | 0.012 | 0.012 | 0.001 |
| KPP | Node black box | 0.047 | 0.058 | 0.073 | 0.059 | 0.013 |
| KPP | Projected node black box | 0.073 | 0.050 | 0.045 | 0.056 | 0.015 |
| KPP | FV-GNO | 0.036 | 0.036 | 0.034 | 0.035 | 0.001 |
| Flat SWE | Node black box | 0.017 | 0.018 | 0.015 | 0.017 | 0.001 |
| Flat SWE | Projected node black box | 0.017 | 0.015 | 0.015 | 0.016 | 0.002 |
| Flat SWE | FV-GNO | 0.012 | 0.012 | 0.012 | 0.012 | $< 0.001$ |

### D.2 Learning effective interfaces instead of using cell traces

To isolate approximation at the observation interval, we supply each reference state independently and predict its next saved frame. The cell-trace control uses the same prescribed equation adapter and fixed geometric factor $\Delta t A/V$, replacing learned interface states with their incident cell states. Both paths apply the same state constraints, conservative conversion, and output conversion. The diagnostic uses one observation-interval update; the classical-solver experiment in Section 5.3 instead allows CFL-limited substeps.

Parameters and normalizers are frozen. Evaluation uses FP32 IEEE eager inference and FP64 metrics. For each configuration, one-step error averages time/volume MSE, takes the square root per channel,

and then averages channels and trajectories. The reported comparisons cover seeds 123/456/789. Mean cell-trace and learned-interface errors are, respectively, $1.034 \times 10^{-2}$ and $7.380 \times 10^{-4}$ for Burgers, $1.282 \times 10^{-1}$ and $4.783 \times 10^{-3}$ for KPP, $6.580 \times 10^{-3}$ and $1.533 \times 10^{-3}$ for flat water, and $4.436 \times 10^{-3}$ and $6.274 \times 10^{-4}$ for topographic water.

The cell-trace/model error ratios span 13.868–14.182, 26.770–26.836, 4.272–4.319, and 7.001–7.141 across the three seeds for these four tasks. Figure 6(left) summarizes all four tasks; Fig. 7 resolves Burgers and flat water by training run. These gains isolate the learned approximation while the equation adapter and balance update remain unchanged.

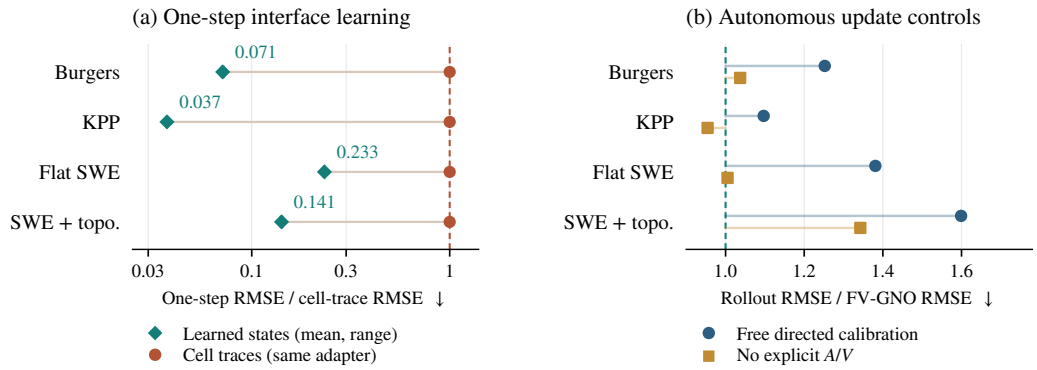


Figure 6: **Interface and geometric ablations.** Left: reference-state one-step error normalized by the cell-trace control, with adapter and geometry fixed. Diamonds are three-seed means and bars span seed minima/maxima. Right: individual-run autonomous errors normalized by the corresponding FV-GNO run, for the controls in Table 12. The local test applies one observation-interval update.

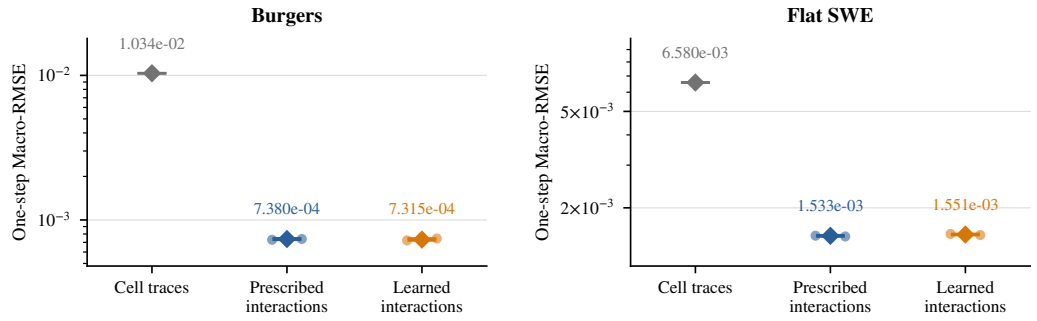


Figure 7: Local approximation for both interface-state and shared-flux parameterizations. Diamonds show mean one-step macro-RMSE, whiskers span the three training seeds, and small points are individual runs. Cell traces are deterministic for the fixed reference data and adapter. Each run first averages reference trajectories. The vertical axes are logarithmic.

The shared-flux ablation also improves substantially on cell traces. Its mean one-step error is $7.315 \times 10^{-4}$ on Burgers and $1.551 \times 10^{-3}$ on flat water, close to the corresponding effective-state errors. The large gain in this local test is therefore from learning the missing approximation; autonomous prediction and equilibrium retention below examine the additional consequences of how that approximation is parameterized.

### D.3 Geometric coefficients and paired assembly

The free-calibration control learns directed multipliers of the update coefficients. The no-explicit-$A/V$ control removes that explicit geometric factor while retaining the other graph geometry inputs

and shared scaling. The cell-traces-plus-calibration control combines incident-cell inputs with learned directed coefficients. These interventions are evaluated as autonomous rollouts in Table 12, rather than on independently supplied reference states.

Known geometry with paired assembly has lower error than free directed calibration on all four regular tasks (Fig. 6, right). The combined cell-trace/calibration control has higher error on the homogeneous tasks and nonfinite topographic predictions. Removing explicit $A/V$ worsens Burgers and both shallow-water tasks, while KPP improves slightly; on a uniform grid, remaining geometry inputs and shared scaling can absorb a constant factor. Directed calibration changes both the coefficients and their volume-weighted pairing, whereas the reference-state interface test above keeps both fixed.

The implemented fixed-geometry rollouts have maximum closed-invariant drift below $3 \times 10^{-7}$. Appendix E.1 checks the executed flux/source assembly and state conversions. The geometry controls therefore examine predictive consequences of the numerical organization in addition to its algebraic balance.

### D.4 A SHARED-FLUX ABLATION WITHIN OUR FINITE-VOLUME FRAMEWORK

**What is changed.** To test the choice of learning interface states, we develop a shared-flux variant of our finite-volume framework as an internal ablation. The prescribed-flux model decodes equation-specific two-sided states and passes them through the physical numerical adapter. The shared-flux model instead produces signed proposals in both directions and constructs

$$\widehat{\Phi}^{\mathrm{L}}_{ij} = \tfrac{1}{2}\big(\psi_\theta(i,j) - \psi_\theta(j,i)\big), \qquad \widehat{\Phi}^{\mathrm{L}}_{ji} = -\widehat{\Phi}^{\mathrm{L}}_{ij}. \tag{41}$$

The regular tasks retain the same fixed factor $\Delta t A/V$ and paired assembly. This changes the learning location while retaining the same regular-grid numerical organization.

**Matched training and equation-specific changes.** Both variants share the data split, graph, processor architecture, normalized MSE, noise schedule, and training budget within each task. Training uses 20 one-step epochs followed by four three-step epochs, independently initialized with seeds 123/456/789. Evaluation restores each model's frozen normalization statistics and uses all test trajectories over the full horizon. Table 14 reports the mean and sample SD of these independently trained models.

For Burgers, KPP, and flat water, both branches have the same parameter count. In Burgers, the shared-flux branch replaces the complete convective–viscous exchange. For KPP it replaces the sine/cosine LLF adapter; for flat water it replaces HLL evaluated on $(h, hu, hv)$. Signed flux outputs change the decoder semantics and its state-specific positivity maps. Topographic water retains the interface-state head, hydrostatic reconstruction, and analytic source corrections, and adds a three-channel flux head with 14,883 parameters: 1,206,315 versus 1,221,198 parameters in total. Its retained velocity-head outputs do not enter the final source correction. Both Euler branches retain directed geometric calibration and the same prescribed boundary sequence; shared flux alone does not pair the directed coefficients. Euler training loss is evaluated on interior NORMAL nodes. At each predicted time the prescribed boundary values are restored before saving; full-field RMSE includes these zero-error boundary nodes, whereas the local-region diagnostics use NORMAL nodes only. Both branches use the same boundary treatment.

Table 14: Prescribed physical flux versus the shared-flux ablation within the finite-volume framework. Macro-RMSE ($\times 10^{-3}$), mean $\pm$ sample SD over seeds 123/456/789. Euler uses directed calibration in both branches.

| Task | Prescribed physical flux | Shared-flux ablation |
|---|---|---|
| Burgers | $12.280 \pm 0.522$ | $11.775 \pm 0.102$ |
| KPP | $35.413 \pm 1.383$ | $36.175 \pm 1.343$ |
| Flat SWE | $11.676 \pm 0.165$ | $11.926 \pm 0.908$ |
| SWE + topo. | $4.402 \pm 0.812$ | $3.553 \pm 0.142$ |
| Euler | $113.623 \pm 14.759$ | $132.614 \pm 23.636$ |

**Dynamic accuracy on regular and irregular grids.** Across the four regular-grid tasks, the prescribed-flux model and our shared-flux ablation achieve comparable absolute macro-RMSE values, with differences below $10^{-3}$ in every case. On irregular Euler meshes, retaining the prescribed physical flux reduces the mean error from $0.132614$ to $0.113623$, while also reducing the across-seed standard deviation. Together with the reference-integral and long-horizon equilibrium diagnostics below, these results motivate our choice to learn effective interface states while retaining prescribed physical fluxes.

### D.5 Physical-flux behavior on irregular Euler observations

For each time, a node's reference jump score is the maximum relative pressure jump on its incident faces, including those adjoining prescribed boundary nodes. We select the top $q \in \{0.05, 0.10, 0.20\}$ fraction of NORMAL nodes, rounding the count up and breaking ties by node index. Region RMSE uses these reference masks; the complement also contains NORMAL nodes only. Front IoU compares top-$q$ sets ranked independently from reference and prediction. Region RMSE pools squared errors over selected nodes and time before taking channelwise roots and their mean; IoU averages over time. Trajectory summaries are then averaged over the 20 cases. Table 15 reports the comparison. In this comparison, the prescribed interaction has lower field error both globally and in the selected high-jump regions.

For conservative variable $Q \in \{\rho, \rho u, \rho v, E\}$, define the relative reference-integral discrepancy

$$I_Q = \frac{1}{H}\sum_{n=1}^{H} \frac{\left|\sum_i V_i(\widehat{Q}_i^n - Q_i^n)\right|}{\max(\sum_i V_i|Q_i^n|, 10^{-12})}, \qquad E = \frac{p}{\gamma - 1} + \tfrac{1}{2}\rho(u^2 + v^2). \tag{42}$$

The sum contains all nodes after restoring prescribed boundaries. Table 16 averages each model's $I_Q$ over 20 test trajectories and then over training seeds 123/456/789. The prescribed interaction lowers the three-seed mean discrepancy by 40.220% for mass and 36.869% for total energy. Momentum discrepancies are similar in the mean. Figure 8 shows the variation across training seeds: mass and energy improve in two of the three matched-seed comparisons. These metrics measure agreement with reference domain integrals under open boundaries; the discrete flux-balance audit appears in Appendix E.1.

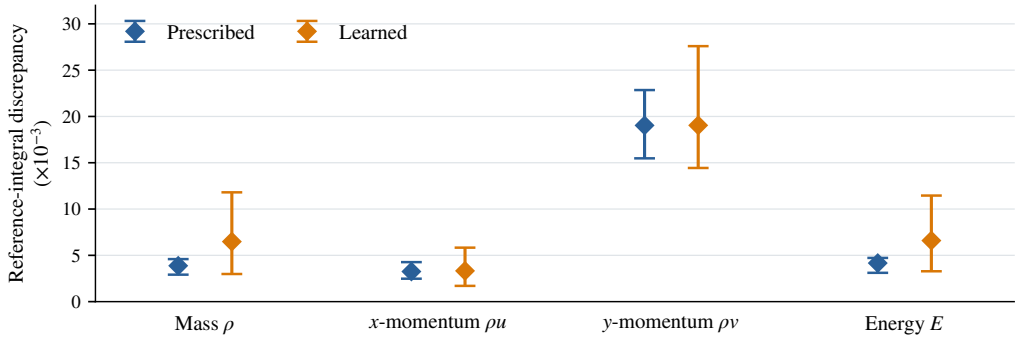


Figure 8: Euler reference-integral discrepancy for prescribed and learned interactions. Diamonds denote the three-seed means; vertical bars span the minimum and maximum seed means, not confidence intervals. Each seed averages the same 20 test trajectories. All four conservative channels are shown.

Table 15: Euler local-region errors for the reported prescribed and shared-flux checkpoint pair. Each value averages 20 trajectories over 79 predicted steps. All-node RMSE includes the prescribed boundary nodes; top-fraction regions and their complements contain only interior NORMAL nodes. Front sets are ranked independently for prediction and reference; region errors use reference-defined masks.

| Region | Metric | Prescribed | Learned |
|---|---|---|---|
| All nodes | Macro-RMSE | 0.101 | 0.160 |
| Top 5% | Macro-RMSE | 0.398 | 0.640 |
| Complement 5% | Macro-RMSE | 0.050 | 0.077 |
| Top 10% | Macro-RMSE | 0.293 | 0.471 |
| Complement 10% | Macro-RMSE | 0.044 | 0.066 |
| Top 20% | Macro-RMSE | 0.211 | 0.339 |
| Complement 20% | Macro-RMSE | 0.043 | 0.063 |
| Top 5% | Front IoU | 0.594 | 0.537 |
| Top 5% | Jump rel. $L^2$ | 0.527 | 0.708 |
| Top 10% | Front IoU | 0.616 | 0.583 |
| Top 10% | Jump rel. $L^2$ | 0.540 | 0.724 |
| Top 20% | Front IoU | 0.638 | 0.635 |
| Top 20% | Jump rel. $L^2$ | 0.547 | 0.735 |

Table 16: Euler reference-integral discrepancy, averaged over training seeds 123/456/789 for each interaction. Each seed first averages the time-normalized discrepancy over all 20 test trajectories. Columns give mass, the two momenta, and total energy. Figure 8 shows the seed ranges. These reference-relative errors under prescribed open boundaries are distinct from a discrete conservation-balance residual.

| Interaction | $\rho$ | $\rho u$ | $\rho v$ | $E$ |
|---|---|---|---|---|
| Prescribed | $3.875 \times 10^{-3}$ | $3.251 \times 10^{-3}$ | $1.903 \times 10^{-2}$ | $4.163 \times 10^{-3}$ |
| Learned | $6.483 \times 10^{-3}$ | $3.323 \times 10^{-3}$ | $1.904 \times 10^{-2}$ | $6.595 \times 10^{-3}$ |

### D.6 Long-horizon lake-at-rest retention

Stationary source–flux balance probes what equation interactions add within the finite-volume framework. Both parameterizations retain regular-grid geometry, paired assembly, hydrostatic reconstruction, and analytic bed-source treatment. The comparison evaluates their trained interface/interaction parameterizations under stationary initial conditions. P2 and P3 denote the two prescribed-adapter checkpoints in this diagnostic, and L denotes its shared-flux checkpoint; these checkpoint labels are distinct from the three-seed dynamic-prediction summary in Table 14.

For each bed $b$, we initialize

$$h_i = \max_j b_j + h_{\min} - b_i, \qquad u_i = v_i = 0, \qquad h_i + b_i = \max_j b_j + h_{\min}. \tag{43}$$

This is a stationary solution of the shallow-water balance law. We summarize 256-step predictions at $\Delta t = 0.01$ for minimum depths $h_{\min} \in \{0.5, 1, 2\}$ on 27 further held-out bed profiles, beyond the original three diagnostic beds. These are the remaining profiles among the same 30 held-out trajectories. Each reported configuration retains positive raw height throughout the rollout, with no output floor activation. The horizon exceeds the training-trajectory duration.

For each bed, accumulated state error through step $H$ is

$$E_{\mathrm{prim}}(H) = \frac{1}{3} \sum_{c \in \{h,u,v\}} \left[ \frac{1}{HN} \sum_{n=1}^{H} \sum_{i} (\widehat{q}^{\,n}_{i,c} - q^{0}_{i,c})^2 \right]^{1/2}. \tag{44}$$

Let $\overline{E}_{\mathrm{P}}$ and $\overline{E}_{\mathrm{L}}$ average this error over the 27 beds for a prescribed-adapter model and the learned model. Their relative reduction is $G = 100(1 - \overline{E}_{\mathrm{P}}/\overline{E}_{\mathrm{L}})$. This measures the improvement relative to the learned interaction at each depth.

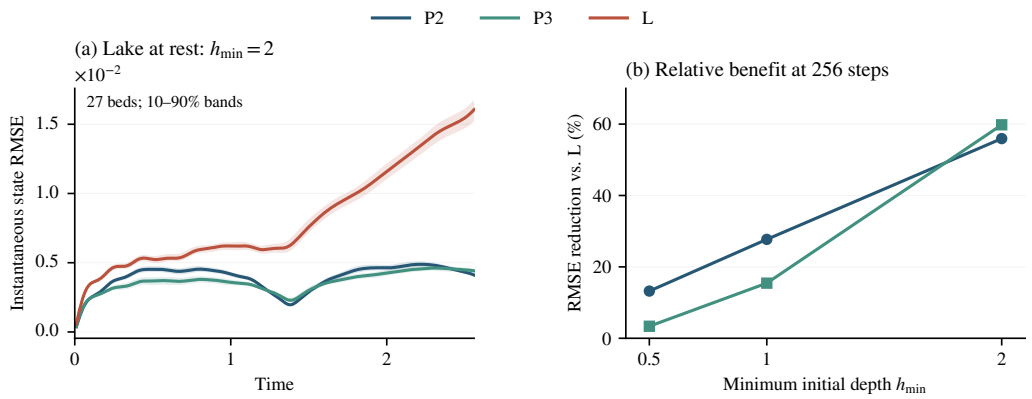


Figure 9: Long-horizon equilibrium retention on 27 further bed profiles. (a) At $h_{\min} = 2$, mean instantaneous state RMSE over 256 steps; bands span the 10th–90th bed percentiles. (b) Relative reduction of accumulated RMSE against L at the three reported depths, for P2/P3. The relative advantage increases with depth for both prescribed-adapter models.

Table 17: Lake-at-rest retention over 256 steps on 27 further bed profiles at three tested minimum initial depths. Errors are $E_{\mathrm{prim}}$ in units of $10^{-3}$, averaging channel RMSEs of $(h, u, v)$ and then all 27 beds. P2/P3 denote the prescribed-adapter checkpoints and L the shared-flux checkpoint. Gain is $100(1 - E_{\mathrm{prim}}(P)/E_{\mathrm{prim}}(L))$ in percent. Every model and bed is retained at each displayed depth.

| $h_{\min}$ | L: $E_{\mathrm{prim}}$ | P2: $E_{\mathrm{prim}}$ | P3: $E_{\mathrm{prim}}$ | P2 gain (%) | P3 gain (%) |
|---|---|---|---|---|---|
| 0.500 | 4.779 | 4.147 | 4.618 | 13.219 | 3.367 |
| 1.000 | 1.353 | 0.978 | 1.144 | 27.707 | 15.443 |
| 2.000 | 9.212 | 4.059 | 3.703 | 55.937 | 59.800 |

**A larger relative advantage at greater depth.** Both prescribed-adapter models have lower mean error than L at each reported depth (Table 17). Their reductions increase from 13.219% and 3.367% at $h_{\min} = 0.5$, to 27.707% and 15.443% at $h_{\min} = 1$, and to 55.937% and 59.800% at $h_{\min} = 2$. At depth 2, both improve on *every one of the 27 beds*. Figure 9(a) shows how this separation develops over time. These results support improved long-horizon equilibrium retention by the prescribed numerical parameterizations in the reported depth range.

**Why retain the prescribed physical flux.** The shared-flux ablation establishes that the finite-volume organization can also support a learned interaction. Our default instead retains the physical numerical adapter: it combines competitive regular-grid prediction with lower Euler mean error, improved Euler mass and energy agreement, and stronger long-horizon equilibrium retention in the tested stationary configurations. The latter two diagnostics test equation-specific behavior beyond the training-horizon dynamic error. Together they motivate learning effective interface states while using the known physical interaction to evaluate their exchanges.

### D.7 Replacing the graph processor

The interface decoder, equation adapter, and update remain the same when the neural-operator processor is replaced with an MPNN. The four regular tasks retain fixed finite-volume geometry; Euler retains its calibrated configuration. These five single-run comparisons show that the finite-volume organization can be used with distinct graph processors.

Table 18: Processor replacement with the same interface and update modules: single-run macro-RMSE. Regular tasks use fixed finite-volume geometry; Euler uses directed calibration.

| Task | Neural-operator processor | MPNN |
|---|---|---|
| Burgers | 0.012 | 0.011 |
| KPP | 0.036 | 0.030 |
| Flat SWE | 0.012 | 0.009 |
| SWE + topo. | 0.004 | 0.004 |
| Euler (calibrated) | 0.101 | 0.106 |

# E BALANCE AND OBSERVATION COMPATIBILITY

We audit the paired regular-grid implementation, then examine geometric compatibility and deterministic execution for the open-boundary Euler extension. The flux-parameterization diagnostics are grouped with the ablations in Appendix D.5.

## E.1 CONSERVATION IN THE IMPLEMENTED UPDATE

The exact identity in Appendix A.1 concerns paired algebraic exchange. We audit its implementation separately from long-rollout agreement with reference integrals. For the homogeneous periodic tasks, the conserved channels are all components of the conservative state. For topographic shallow water, mass is conserved, and momentum changes include the prescribed bed force.

Two measurements answer different questions. Let $\langle Z\rangle_V = (\sum_i V_i)^{-1} \sum_i V_i Z_i$. The closed-invariant drift is

$$D = \max_{k,t,c\in\mathcal{I}} \frac{|\langle U^t_{k,c} - U^0_{k,c}\rangle_V|}{\max(\langle |U^0_{k,c}|\rangle_V, 10^{-12})}, \tag{45}$$

where $\mathcal{I}$ contains all conservative channels for the homogeneous tasks and only mass for topographic shallow water. This checks the actual stored rollout, including conversion and floating-point accumulation.

The second check compares the raw conservative update with its directed cell-update messages $g_e$. With $V_\Omega = \sum_i V_i$,

$$r_{\mathrm{asm}} = \left| \langle U_{\mathrm{raw}} - U_{\mathrm{in}}\rangle_V + V_\Omega^{-1} \sum_e g_e \right|, \tag{46}$$

$$b_{\mathrm{fp}} = 64\epsilon_{\mathrm{FP32}} \left( \langle |U_{\mathrm{raw}}| + |U_{\mathrm{in}}|\rangle_V + V_\Omega^{-1} \sum_e |g_e| \right) + 10^{-30}. \tag{47}$$

Here the stored $g_e$ includes the volume weighting used in this sum. The table reports the largest $r_{\mathrm{asm}}/b_{\mathrm{fp}}$ over the evaluated steps and channels. This checks the implemented update against its flux/source assembly; paired cancellation itself follows from the shared face construction. Agreement with a reference trajectory's integral is a separate accuracy diagnostic and is not used as the conservation test.

Table 19: Fixed finite-volume rollout audit. Drift is the maximum recorded closed-invariant drift; the update-assembly residual is normalized by the evaluator's floating-point roundoff budget. Maxima cover all available runs and trajectories. No input height-clamp cell steps occur in these rollouts. Error quantities use uniform scientific notation.

| Task | Max. drift | Assembly / roundoff | Input clamp steps |
|---|---|---|---|
| Burgers | $6.966 \times 10^{-9}$ | $5.947 \times 10^{-5}$ | 0 |
| KPP | $7.131 \times 10^{-9}$ | $8.186 \times 10^{-5}$ | 0 |
| Flat SWE | $2.799 \times 10^{-7}$ | $6.872 \times 10^{-5}$ | 0 |
| SWE + topo. | $3.687 \times 10^{-9}$ | $8.276 \times 10^{-5}$ | 0 |

### E.2 Geometry and pairing for irregular observations

Euler supplies vertex values rather than averages over reconstructed dual cells. We compare dual-cell geometry and point-estimated geometry with fixed or calibrated update coefficients. The common-FP32 experiment shares its full training protocol and deterministic GPU evaluation on 20 trajectories; a separate historical pair isolates forced output pairing under its own matched protocol.

Table 20: Euler geometry controls. The first block shares the full FP32 training protocol and deterministic GPU evaluation on 20 trajectories. A dash marks the explosive point-geometry control. The second block uses a separate matched historical training protocol.

| Geometry/update | Macro-RMSE |
|---|---|
| Dual geometry, fixed | 0.147 |
| Dual geometry, calibrated | 0.270 |
| Point geometry, fixed | - |
| Point geometry, calibrated | 0.281 |
| Separate historical pairing protocol | |
| Directed calibration | 0.130 |
| Forced output pairing | 0.647 |

Fixed dual geometry is best in the common-FP32 block (0.147), whereas forcing output pairing raises error from 0.130 to 0.647 in the historical pair. Geometry and pairing should therefore match the observation representation. These mechanism controls use normalization and precision protocols different from the final prediction recipe; each comparison is interpreted within its own matched block.

### E.3 Deterministic Euler evaluation

The geometry comparison in Table 20 evaluates the geometry alternatives with a common deterministic GPU implementation. It uses IEEE FP32, disables TF32, enables PyTorch deterministic algorithms, and sets `CUBLAS_WORKSPACE_CONFIG=:4096:8` before CUDA initialization. Each evaluation uses all 20 test trajectories and 79 updates, with the original initial states, boundary values, time interval, and equation adapter. No weights, normalization statistics, or physical formulas are changed. Fixed dual geometry is evaluated three times; the other geometry controls are evaluated once under the same execution settings.

**Separating execution variability from training variability.** The fixed-dual model originally scored $0.150$. Three archived evaluations of those same weights gave full-test errors ranging from $0.146$ to $0.318$. These are repeated executions of one model, distinct from the independently trained seeds in the prediction benchmark. The discrepancy is concentrated in a sensitive trajectory: case 06 contributes $0.173$ to the $0.172$ increase between the first and third archived full-test means. Under deterministic evaluation, all three full-test errors are $0.147$.

**A controlled aggregation test.** We capture the first graph layer's message and index tensors and repeat its mean aggregation 20 times with identical inputs. Default CUDA aggregation produces 20 distinct outputs, with a maximum absolute difference of $4.768 \times 10^{-7}$. Deterministic aggregation

Table 21: Same-weight execution audit for fixed dual geometry. Repeats hold the trained model fixed and measure execution variation. The first two rows cover the complete test set; the last two isolate the sensitive trajectory under a controlled execution switch. Maximum state difference is over all nodes, channels, and 79 steps relative to repeat one; the archived full-test audit did not retain this field.

| Scope | Execution | Repeat 1 | Repeat 2 | Repeat 3 | Max. difference |
|---|---|---|---|---|---|
| Full test (20) | Archived default | 0.146 | 0.150 | 0.318 | - |
| Full test (20) | Deterministic | 0.147 | 0.147 | 0.147 | 0.000 |
| Case 06 | Default | 0.251 | 3.488 | 0.249 | 3053.131 |
| Case 06 | Deterministic | 0.250 | 0.250 | 0.250 | 0.000 |

produces one output with zero difference. The installed graph library implements this operation through a scatter reduction. This intervention directly identifies a source of execution variability; other reductions in the update are not individually isolated. In the case-06 rollout control, both groups use the same CUDA workspace configuration and differ only in the deterministic-algorithm setting. The deterministic group reproduces all 79 predicted states exactly.

**Why a small perturbation can matter.** The sensitive trajectory first develops nonpositive pressure at step 9 and nonpositive density at step 13. The conservative-to-primitive map contains $u = m_x/\rho$ and a kinetic-energy term proportional to $\|m\|^2/\rho$; small density perturbations can therefore be strongly amplified near zero density. The model's input floor acts on the next update, after this conversion. This mechanism is consistent with the observed growth from aggregation roundoff to large local state differences. Deterministic execution restores repeatability; it does not impose positivity. The archived evaluations and the controlled replay records are preserved separately from the new geometry results.

# F Resolution transfer and training-noise studies

This appendix expands the $64^2 \to 128^2$ adaptation experiment in Section 5.4 and separately reports training-noise choices for the directed-calibration variant.

## F.1 Few-shot adaptation from $64^2$ to $128^2$

Both the $64^2$ source models and the $128^2$ target models use fixed finite-volume geometry and paired assembly. Source pretraining uses split A; target adaptation uses nested subsets of split B containing 1, 4, 16, or all trajectories. Splits A/B/C contain 135/135/30 trajectories for Burgers and topography, and 140/140/20 for KPP and flat water. Scratch and pretrained models receive the same target data, frozen source normalization, and target optimization budget; testing uses the disjoint split C. The ratio is pretrained error divided by scratch error. Each entry reports the result for its target-data setting; ratios are retained from the recorded evaluation before rounding the displayed errors. Table 22 evaluates transfer of the learned interface representation while retaining the numerical update at both resolutions. On Burgers, KPP, and flat water, pretraining reduces error at every target-data budget by 41–66% relative to scratch.

The 16-trajectory results are highlighted in the main text because they also beat full-split scratch training on all three homogeneous tasks. Across the complete sweep, the benefit persists from one target trajectory through the full target split. For topographic water, the few-shot runs are explosive; only the full-data pretrained model gives a usable score (0.008), while its scratch counterpart remains unusable. This task is retained in the full table.

Table 22: $64^2 \rightarrow 128^2$ adaptation with fixed finite-volume geometry and paired assembly in both source and target models. Entries are macro RMSE; the ratio is pretrained/scratch. Source pretraining uses split A; target adaptation and scratch training use identical split-B trajectories, frozen source normalization, and matched target optimizer steps. Evaluation uses all of split C. Full denotes all of B. Dashes mark explosive finite topographic rollouts; the ratio is omitted if either result is unusable.

| Task | Target trajectories | Scratch | Pretrained | Ratio |
|---|---|---|---|---|
| Burgers | 1 | 0.090 | 0.041 | 0.457 |
| Burgers | 4 | 0.065 | 0.023 | 0.359 |
| Burgers | 16 | 0.046 | 0.017 | 0.372 |
| Burgers | full | 0.028 | 0.016 | 0.565 |
| KPP | 1 | 0.427 | 0.238 | 0.559 |
| KPP | 4 | 0.266 | 0.110 | 0.412 |
| KPP | 16 | 0.146 | 0.049 | 0.339 |
| KPP | full | 0.105 | 0.043 | 0.407 |
| Flat SWE | 1 | 0.225 | 0.109 | 0.484 |
| Flat SWE | 4 | 0.100 | 0.057 | 0.568 |
| Flat SWE | 16 | 0.043 | 0.016 | 0.365 |
| Flat SWE | full | 0.022 | 0.013 | 0.586 |
| SWE + topo. | 1 | - | - | - |
| SWE + topo. | 4 | - | - | - |
| SWE + topo. | 16 | - | - | - |
| SWE + topo. | full | - | 0.008 | - |

### F.2 Training-noise settings

The topographic shallow-water study changes additive noise during training and evaluates all models from clean initial states (Bishop, 1995). It includes state-wise low/high noise and absolute Gaussian standard deviations of $10^{-3}$ and $10^{-2}$. The low/high vectors are also absolute physical-unit scales in channel order $(h, u, v)$. The default recipe uses relative noise with a floor. This archived study uses the directed-calibration variant; its results are kept separate from the fixed finite-volume accuracy table.

Table 23: Training-noise study on topographic shallow water with directed calibration. All evaluations use 30 clean test initial states. Absolute standard deviations follow $(h, u, v)$; the default relative recipe changes between training stages.

| Setting | Training $\sigma$ | $h$ | $u$ | $v$ | Macro |
|---|---|---|---|---|---|
| Default recipe | 0.010 to 0.002 | 0.007 | 0.007 | 0.006 | 0.007 |
| Absolute scalar 0.001 | (0.001, 0.001, 0.001) | 0.007 | 0.007 | 0.007 | 0.007 |
| Absolute scalar 0.010 | (0.010, 0.010, 0.010) | 0.011 | 0.010 | 0.010 | 0.010 |
| State-wise low | (0.005, 0.002, 0.001) | 0.009 | 0.008 | 0.008 | 0.008 |
| State-wise high | (0.050, 0.020, 0.010) | 0.017 | 0.016 | 0.015 | 0.016 |

## G PREDICTION FIELDS

Figures 10–14 compare reference and predicted states at four saved times. Regular cases use the fixed finite-volume models reported in the main comparison; Euler uses the calibrated production extension. Figures use the stored predictions; source identities and times are recorded in the experimental archive.

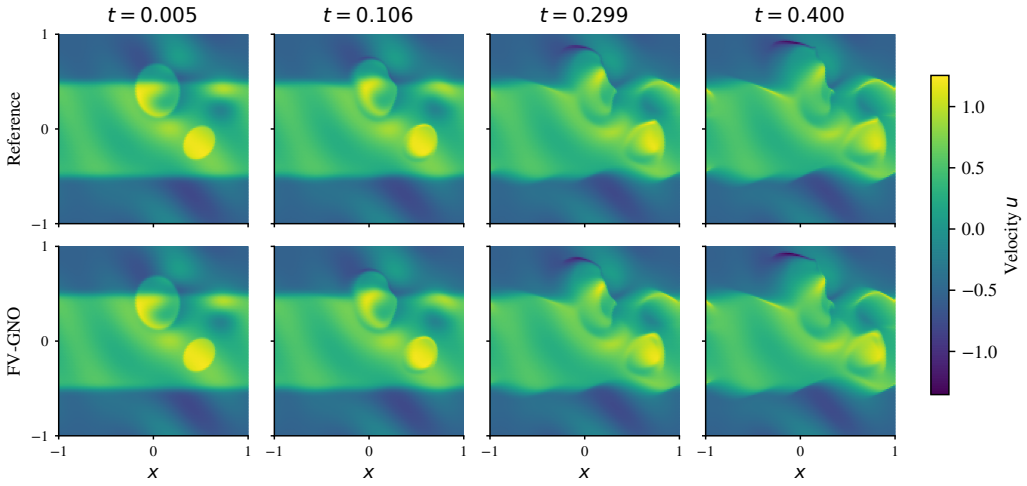


Figure 10: Conservative vector viscous Burgers: reference and predicted velocity at four times.

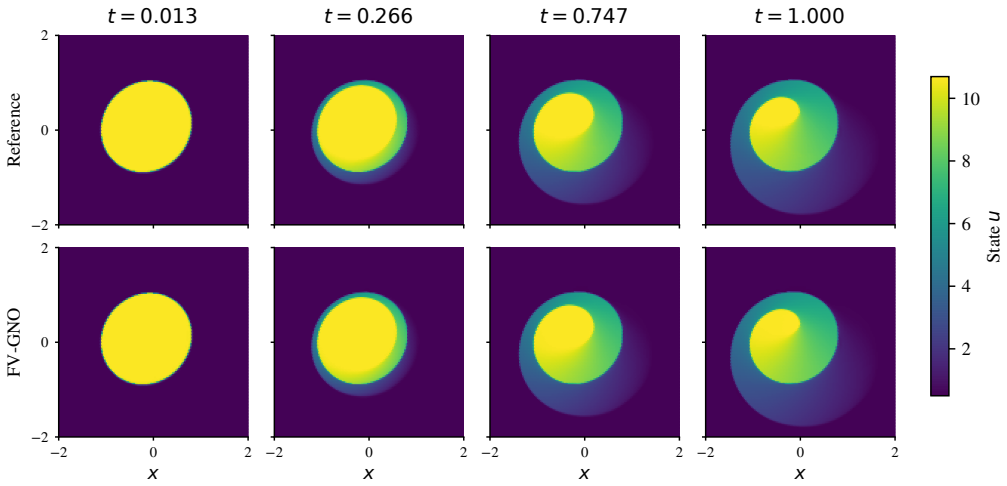


Figure 11: KPP conservation law: reference and predicted state at four times.

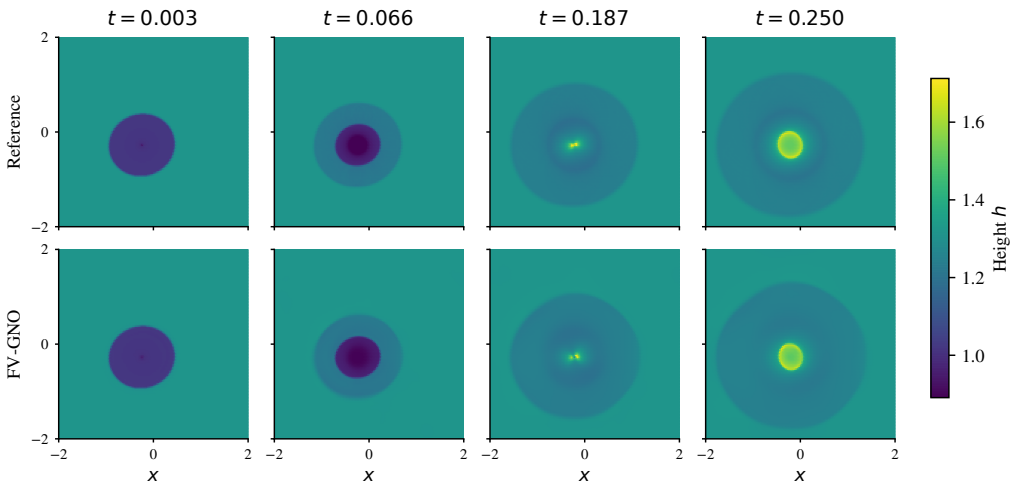


Figure 12: Flat shallow water: reference and predicted height at four times.

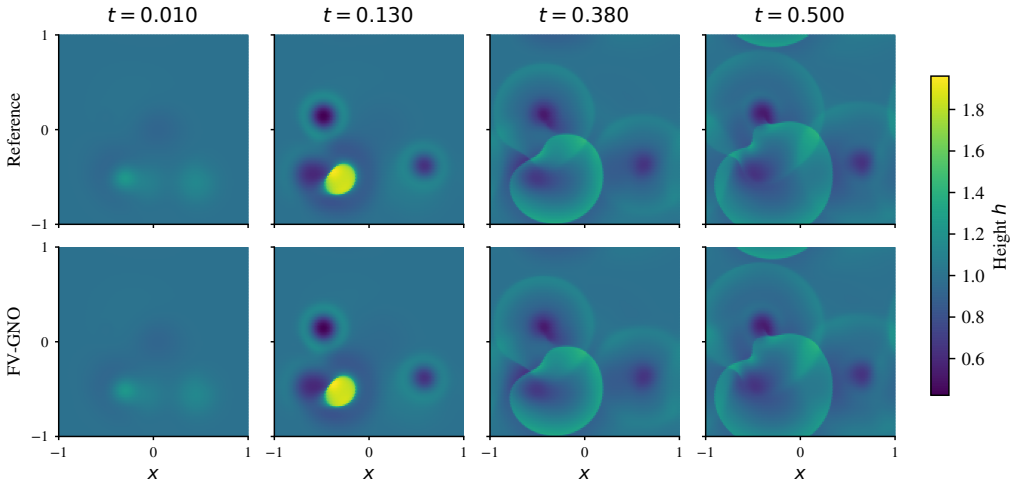


Figure 13: Topographic shallow water: reference and predicted height at four times.

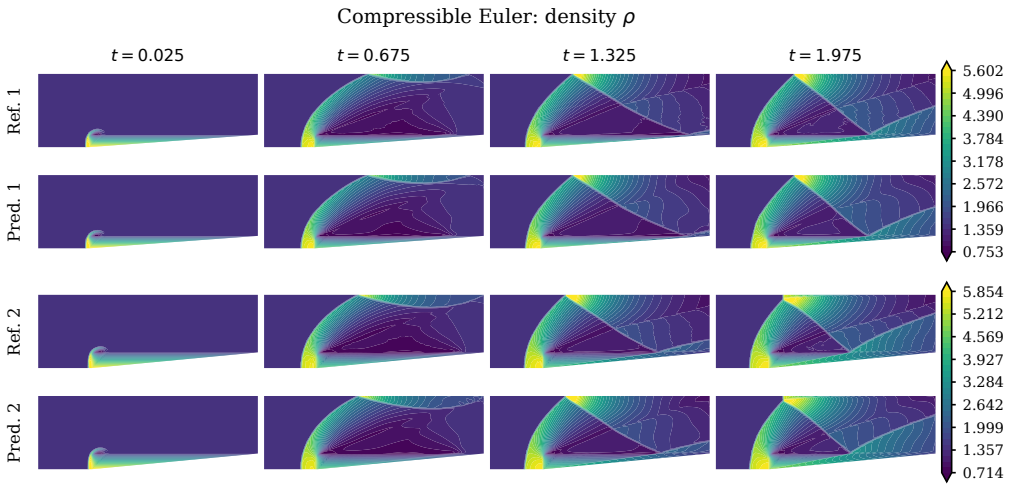


Figure 14: Irregular Euler density: reference and calibrated-model predictions at four times for two trajectories.