CROSS-DOMAIN PRETRAINING FOR STEADY-STATE NEURAL CFD SURROGATES

Anonymous authors
Paper under double-blind review

ABSTRACT

Neural surrogates for computational fluid dynamics (CFD) have the potential to greatly enhance engineering innovation through accelerating simulation. However, the primary limitation for neural surrogates is the lack of generalization to geometries and applications beyond the training set, which is significant given the diversity of engineering scenarios. Currently, this is addressed by generating a new dataset for a specific application; however, this requires running costly numerical solvers. In this work, we take a step toward addressing this by studying neural surrogates trained across different geometries, boundary conditions, and fidelities. We find that cross-domain pretraining improves zero- and few-shot performance on held-out datasets relative to both training from scratch and transferring from domain-specific experts. In particular, finetuning a pretrained, cross-domain model can achieve 2-3x lower errors at the same sample size and use 8x fewer samples to achieve the same error, compared to training from scratch. This benefit is architecture agnostic and improves with model size and pretraining dataset diversity. Furthermore, we study how and why cross-domain pretraining works in CFD surrogates, and find that simply pooling steady-state datasets is both sufficient and effective. Given the high cost of generating CFD data, leveraging existing datasets through cross-domain pretraining will likely be a valuable strategy as future surrogates expand to tackle new problems and use cases.

1 INTRODUCTION

Computational fluid dynamics (CFD) is an essential tool in modern engineering, producing simulations that inform design decisions across the automotive, aerospace, and consumer products industries. Although CFD now supports design and analysis throughout the engineering industry, the simulations themselves remain slow and expensive due to the difficulty of accurately resolving turbulent flows. Recent work in neural PDE surrogates has looked to address this through approximating solver outputs in a data-driven manner. These neural surrogates have been well-studied on uniform-grid, time-dependent PDEs (Ohana et al., 2025; McGreivy & Hakim, 2024) and are now showing promise on 3D, mesh-based CFD problems.

Given sufficient training data, advances in mesh-based transformer models now allow surrogates to accurately predict steady-state surface and volume fields for a geometry and set of simulation inputs (Luo et al., 2025; Alkin et al., 2025a; Hagnberger & Niepert, 2026). These models also reproduce aggregate quantities such as C_d and C_l and support design optimization on practically relevant datasets (Alkin et al., 2025b; Paischer et al., 2026; Thumiger et al., 2026). However, these capabilities are largely limited to the application covered by the training set, whereas engineering practice spans a far wider space of geometries, boundary conditions, and solver fidelities.

Therefore, when deploying a surrogate model to a new application, we currently need to generate a new dataset. This can be very costly and requires a large investment in running the numerical solver the surrogate is meant to replace (Ashton et al., 2025b). Furthermore, neural surrogates unlock the most benefit when approximating high-fidelity simulations, yet these are the most difficult to generate. A recent aerospace dataset from Ashton et al. (2026) required 8 GB200 GPU nodes per sample, putting the estimated cost of its $\sim 10^3$ samples at millions of dollars of compute. Therefore, while we have methods to train CFD surrogates, the limiting factor is their lack of generalizability to new geometries and use cases and the high cost of generating data for a new application.

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.

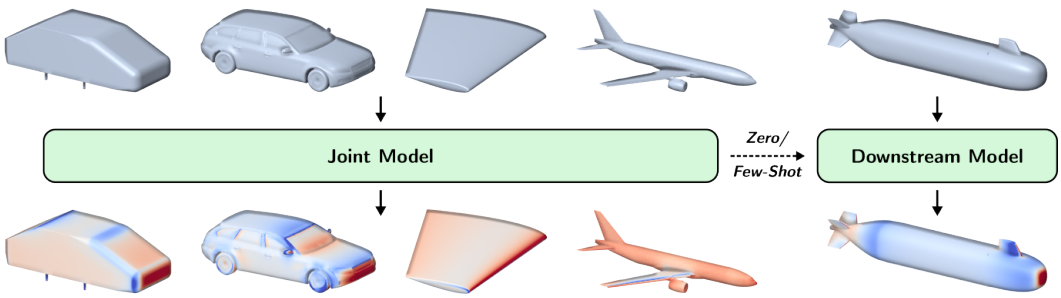

Figure 1: After pretraining a joint model across 6 CFD datasets, we evaluate its zero/few-shot performance on 4 held out datasets (see Table 1).

In this work, we propose mitigating the need for application-specific data by leveraging existing CFD datasets through cross-domain pretraining. We aggregate a diverse set of steady-state, mesh-based CFD datasets, pretrain a set of models, and examine the downstream performance on held-out datasets. We present the first study of cross-domain pretraining for 3D, mesh-based CFD surrogates and make the following contributions:

1. We find that a joint model pretrained across CFD domains outperforms single-domain experts in zero- and few-shot transfer on a variety of held-out datasets.
2. Finetuning the joint model reaches a target error on a held-out dataset with 8–16× fewer training samples compared to training from scratch.
3. We show that finetuning performance on unseen datasets improves with both pretraining dataset diversity and model size.
4. We release a unified collection of 8 publicly available CFD datasets, converted to a common format, along with dataloaders and training code for cross-domain experiments.

2 Related Works

Cross-Domain Training for PDEs. Several works have considered training large-scale neural surrogates across uniform-grid, time-dependent PDEs. This is largely driven by robust data infrastructure and benchmarks (Ohana et al., 2025; Takamoto et al., 2024; Subramanian et al., 2023), as well as the speed of uniform-grid spectral methods used for data generation (Koehler et al., 2024; Liu et al., 2026). These foundation models (McCabe et al., 2024; Hao et al., 2024; Herde et al., 2024; Sun et al., 2025; Morel et al., 2025; Ye et al., 2026; Rautela et al., 2026; McCabe et al., 2026) demonstrate good performance within their pretraining set, improved finetuning performance on held-out datasets, and can generalize to challenging, experimental applications (Mukhopadhyay et al., 2026).

In addition to leveraging scale, previous works have developed specific methods for cross-domain learning in PDEs, such as meta-learning (Kirchmeyer et al., 2022; Blanke & Lelarge, 2024; Koupaï et al., 2024), in-context learning (Yang et al., 2023; Serrano et al., 2025; Jiao et al., 2025), or scaling test-time compute (Serrano et al., 2026; Mansingh et al., 2026). Prior work has also explored self-supervised approaches, such as JEPA (Assran et al., 2023) and masked reconstruction (He et al., 2021), for learning representations across PDE datasets (Zhou & Farimani, 2024; Chen et al., 2025; Zhou et al., 2024; Qu et al., 2026).

Mesh-based CFD Surrogates. While uniform grids are a good test bench and are used in certain applications (such as weather (Rasp et al., 2024)), the majority of engineering applications solve fluid dynamics on an unstructured mesh. Therefore, several prior works have developed surrogate models that can operate on meshes, such as using GNNs (Pfaff et al., 2021; Lino et al., 2025; Lam et al., 2023), transformers (Li et al., 2023a; Hao et al., 2023; Wu et al., 2024), continuous convolutions (Li et al., 2023c; Hagnberger et al., 2025; Ranade et al., 2025; Koupaï et al., 2025), or latent spaces (Li et al., 2023b; Wang & Wang, 2024; Zhou et al., 2025; Li et al., 2025a;b). As these methods have matured, mesh-based surrogates are increasingly being applied to harder problems, where

Name	Geometry	Closure	Solver	# Samples	# Cells	Re #	Inlet (m/s)	AoA (°)
DrivAerNet++	DrivAer Car	RANS κ-ω	OpenFOAM	8000	24M	8e6	30	-
DrivAerML	DrivAer Car	SA-DDES	OpenFOAM	500	160M	7.2e6	38.9	-
WindsorML	Windsor Body	WMLES	Volcano	355	280M	2.9e6	40	-
AhmedML*	Ahmed Body	SA-DDES	OpenFOAM	500	20M	7.7e5	1	-
SHIFT-Sub*	Submarine	RANS	Luminary	100	32M	3.2e7	5	-
Emmi Wing	Tapered Wings	RANS SA	OpenFOAM	30,000	3.3M	5-20e6	[150, 300]	[-10, 10]
SuperWing	Kinked Wings	RANS SA	ADflow	28,856	3.1M	2e7	[260, 313]	[2, 12]
Double-Delta	Delta Wings	RANS SA	SU2	2448	7M	8e7	101.5	[11, 19]
HiLiftAeroML*	Passenger Plane	WMLES	charLES	1800	500M	1.6e6	68	[4, 22]
SHIFT-CCA*	Combat Aircraft	RANS	Luminary	100	32M	3.1e7	212.5	2

Table 1: Datasets considered in this work, roughly split into automotive (top) and aerospace (bottom) applications, spanning a variety of geometries, fidelities and physical regimes. Datasets with an asterisk* are held-out for downstream evaluation.

the main challenge is the sheer number of mesh points. Engineering CFD simulations are typically 3D and finely discretized, owing to the high Reynolds numbers ($10^5 - 10^7$) and complex geometries of industrial flows. As a result, these meshes can exceed 100 million cells, and only recently have methods emerged that can predict solution fields at this scale. Current state-of-the-art surrogates generally rely on some form of efficient attention, either through sub-sampling (Alkin et al., 2025a; Hagnberger & Niepert, 2026; Giral et al., 2026), local attention (Zhdanov et al., 2025), or sparse attention (Curvo et al., 2026; Zhdanov et al., 2026; Zhou et al., 2026). It is also important to be able to predict high-resolution fields sequentially, such that the memory can scale sub-linearly with the number of mesh points. Most related to our current work is a transfer learning study from Keum & Warey (2026), which examines transferring CFD surrogates within a single domain and family of geometries (SUVs). In contrast, we pretrain across multiple domains (e.g., automobiles and aircraft), show that cross-domain pretraining outperforms single-domain experts, and study transfer and scaling behavior on domains unseen during pretraining.

3 METHODS

Background. We consider the task of steady-state field prediction. Given a set of query points $x = \{x_i\}_{i=1}^N$ with $x_i \in \mathbb{R}^3$, the goal is to predict the corresponding field values $f(x) \in \mathbb{R}^{N\times d}$. These values depend on the geometry $G = \{g_j\}_{j=1}^M$ of an object immersed in the flow, represented as a point cloud with $g_j \in \mathbb{R}^3$, and on a vector of simulation parameters $\theta \in \mathbb{R}^c$ (e.g., inlet velocity or angle of attack). We approximate this mapping with a surrogate model f_ϕ with parameters ϕ:

$$f_\phi : \mathbb{R}^{N\times 3} \times \mathbb{R}^{M\times 3} \times \mathbb{R}^c \to \mathbb{R}^{N\times d}, \qquad f_\phi(x;\, G, \theta) \approx f(x;\, G, \theta), \quad i = 1, \ldots, N,$$

where each prediction depends globally on the geometry G, the simulation parameters θ, and potentially on the other queries in x. In general, N can be large (up to 10^8 or more), while M can be controlled by sampling the raw geometry. It is desirable that the surrogate model $f_\phi(x; G, \theta)$ does not depend on the discretization of the geometry or the queries, and that the solution field can be predicted sequentially in batches of size $n < N$ when the full resolution is needed.

The field values are divided into surface and volume fields, depending on where the query lies. In particular, we consider $\mathcal{S} \subset \mathbb{R}^3$ as the surface of the geometry, from which the point cloud G is sampled, and $\Omega \subset \mathbb{R}^3$ as the surrounding fluid volume. This results in the standard surface and volume fields from the Navier-Stokes equations: surface pressure p and wall-shear stress $\vec{\tau}_w$ on $\mathcal{S}$, and pressure p and velocity $\vec{u}$ in Ω. The target field is then

$$f(x;\, G, \theta) = \begin{cases} f_s(x;\, G, \theta), & x \in \mathcal{S}, \\ f_v(x;\, G, \theta), & x \in \Omega. \end{cases}, \quad f_s = (p, \vec{\tau}_w) \in \mathbb{R}^4, \quad f_v = (p, \vec{u}) \in \mathbb{R}^4 \tag{1}$$

so that $d = 4$ for both fields. These fields are both important for CFD. Surface fields can be integrated to obtain the aerodynamic forces on the object and derive drag and lift, while volume fields can show how geometric features affect the surrounding flow and indicate regions for improvement.

Datasets. To study cross-domain learning for CFD surrogates, we combine K datasets into a single training set $\mathcal{D} = \mathcal{D}_1 \cup \cdots \cup \mathcal{D}_K$ and sample from it, optionally weighting each dataset $\mathcal{D}_k$ by a sampling probability p_k, with $\sum_{k=1}^{K} p_k = 1$. We consider 10 steady-state CFD datasets, split into 6 pretraining sets and 4 fine-tuning sets.

The pretraining sets are: DrivAerNet++ (Elrefaie et al., 2025), DrivAerML (Ashton et al., 2025c), WindsorML (Ashton et al., 2025a), Emmi Wing (Paischer et al., 2026), SuperWing (Yang et al., 2026), and Double-Delta (Shen & Alonso, 2026). These cover common automotive and aerospace applications and span a diverse set of physical and numerical regimes. For example, the solutions are computed with both steady and unsteady solvers, using scale-resolving and time-averaged (RANS) approaches, and for incompressible and compressible flows. The main details of each dataset are shown in Table 1 to visualize the overall diversity.

The finetuning sets are chosen to have meaningfully different geometries and simulation setups, and are AhmedML (Ashton et al., 2024), HiLiftAeroML (Ashton et al., 2026), SHIFT-Submarine (Luminary Cloud, 2025b), and SHIFT-CCA (Luminary Cloud, 2025a). We ensure that new, unseen physical regimes are present during finetuning, such as high-resolution aerodynamics (HiLiftAeroML) and hydrodynamics (SHIFT-Submarine). Lastly, the datasets differ in sign conventions, storage formats, and metadata, so we post-process each one into a common format optimized for fast dataloading. Additional details and visualizations for each dataset can be found in Appendix A.

Examining the datasets (Figures 6 and 7) suggests why cross-domain training may be beneficial: many physical phenomena are shared across CFD simulations regardless of the specific geometry or flow configuration. For example, flow stagnates at the front of each geometry, producing high pressure and low wall-shear stress. Volumetric pressure and velocity follow a roughly inverse relationship outside of the boundary layer and wake, where regions of high pressure correspond to low velocity and vice versa. A low-velocity wake typically also forms downstream of the trailing edge.

Normalization. Since datasets can span many physical regimes, the raw field values can span many orders of magnitude. We address this by predicting the non-dimensional quantities (coefficient of pressure C_p, skin friction coefficient C_f, and non-dimensional velocity u^*):

$$C_p = (p - p_\infty)/q_\infty, \quad C_f = \vec{\tau}_w/q_\infty, \quad u^* = \vec{u}/U_\infty, \quad q_\infty = \frac{1}{2}\rho_\infty U_\infty^2$$

where ρ_∞, p_∞, U_∞, and q_∞ are the freestream density, pressure, velocity magnitude, and dynamic pressure. After non-dimensionalization, the values are normalized by dataset-specific means and standard deviations. The points in the input geometry G are normalized by subtracting their centroid and scaling by a dataset-specific factor, computed such that the average geometry in a dataset is bounded by a 2 x 2 x 2 box. The queries x are also oriented to the same frame by subtracting the centroid and scaling by the same factor. Geometries and output fields are aligned such that the inlet is at the $-x$ direction, and the $+z$ direction points upwards. Simulation parameters θ are defined as the Mach number and angle of attack ([Ma, AoA]), and are normalized based on global min/max values across all datasets; in automotive datasets, these values are not used and are set to 0.

Models. The primary architecture we use is SMART (Hagnberger & Niepert, 2026), due to its state-of-the-art performance on steady-state CFD tasks. However, we also evaluate architectures (AB-UPT (Alkin et al., 2025a) and Transolver++ (Luo et al., 2025)) to verify that the results are model-agnostic. Given the input geometry G, simulation parameters θ and queries x, each model f_ϕ is trained to predict volume and surface fields $f(x)$ using a mean-squared error $||f_\phi(x; G, \theta) - f(x; G, \theta)||_2^2$. During training, we use a batch size of 1 and sample 32,768 surface and volume query points per batch. To ensure a fair comparison, all models are pretrained with identical settings unless otherwise noted: 32M parameters, 100k gradient steps, and the same learning rate and optimizer. For cross-domain pretraining, each batch is sampled from one of the six pretraining datasets without any extra information such as the dataset index. Note that the joint model therefore sees each dataset around 6 times less often than its corresponding experts. Further details on the architectures, training, and hyperparameters can be found in Appendix B.

Evaluation. We use the Normalized RMSE (NRMSE) to evaluate our models. Each model predicts four surface fields and four volume fields at a fixed set of 32,768 randomly sampled surface and volume points, and the error is computed separately for each quantity: $C_p^{\text{surf}}, C_f, C_p^{\text{vol}}, u^*$. For

Held-out Datasets:	AhmedML				SHIFT-Sub				SHIFT-CCA				HiLiftAeroML			
# FT Samples:	0	2	4	8	0	2	4	8	0	2	4	8	0	2	4	8
No Pretraining																
Random init.	0.68	0.54	0.49	0.34	0.64	0.39	0.33	0.33	0.68	0.23	0.23	0.23	**0.81**	0.63	0.64	0.63
Domain Experts																
DrivAerNet++	0.53	0.18	0.15	0.12	0.61	0.22	0.13	0.10	0.73	0.15	0.11	0.11	1.10	0.53	0.47	0.46
Emmi-Wing	0.73	0.21	0.21	0.19	0.66	0.24	0.19	0.16	0.61	0.14	0.12	0.12	1.10	0.52	0.48	0.49
SuperWing	0.82	0.24	0.22	0.19	0.78	0.23	0.16	0.14	0.74	0.13	0.12	0.12	1.26	0.50	0.46	0.45
DrivAerML	0.51	0.18	0.18	0.16	0.61	0.20	0.12	**0.09**	0.71	0.14	0.11	**0.10**	0.99	0.55	0.46	0.46
WindsorML	0.38	0.17	0.13	0.11	0.59	0.21	0.15	0.11	0.75	0.17	0.14	0.14	1.04	0.57	0.51	0.50
Double-Delta	0.71	0.19	0.19	0.17	0.65	0.24	0.20	0.14	0.69	**0.11**	0.11	0.11	0.86	0.48	0.37	0.36
Cross-Domain																
Joint Model	**0.31**	**0.12**	**0.10**	**0.09**	**0.52**	**0.14**	**0.10**	**0.09**	**0.52**	**0.11**	**0.10**	**0.10**	0.90	**0.46**	**0.34**	**0.33**

Table 2: Validation error on four held-out target datasets. Rows indicate the model initialization; single-domain experts are identified by their pretraining dataset. Columns give the number of finetuning samples, with 0 denoting zero-shot evaluation. Green and red indicate the best and worst single-domain experts, respectively, and bold indicates the lowest overall error.

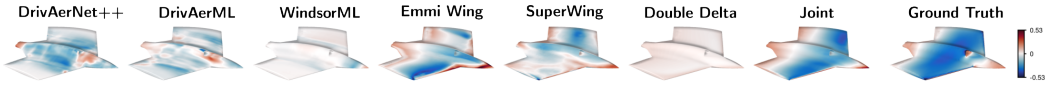

Figure 2: Zero-shot predictions (C_p) of pretrained models on a SHIFT-CCA geometry. Expert models can overfit to their pretraining sets, while the Joint model better predicts new geometries.

brevity, these are shown in Appendix C.6, and in the main results the error is reported as the average over the 4 variable-level errors. Given a predicted field $\hat{\mathbf{u}}$ with n points and c channels ($\hat{\mathbf{u}} \in \mathbb{R}^{n \times c}$):

$$\mathcal{L}_{\text{NRMSE}}(\hat{\mathbf{u}}, \mathbf{u}) = \frac{\|\hat{\mathbf{u}} - \mathbf{u}\|_2}{\|\mathbf{u}\|_2}, \quad \mathcal{L}_{\text{avg}} = \frac{1}{|\mathcal{U}|} \sum_{\mathbf{u} \in \mathcal{U}} \mathcal{L}_{\text{NRMSE}}(\hat{\mathbf{u}}, \mathbf{u}), \quad \mathcal{U} = \{C_p^{\text{surf}}, C_f, C_p^{\text{vol}}, u^*\}.$$

All models are evaluated on validation sets that contain geometries unseen during training. Lastly, all reported errors are computed in the denormalized, physical space (raw C_p, C_f, u^*).

4 Results

4.1 Generalization to Unseen Datasets

Zero/Few-Shot Performance. The primary goal of this work is to leverage prior knowledge from existing CFD datasets to reduce the need for new data when deploying a surrogate to a new engineering task. Therefore, we consider pretraining a joint model across CFD domains, as well as domain experts on each individual dataset. After pretraining, we finetune each model to a restricted number of samples $S \in \{2, 4, 8\}$ with a budget of 400 gradient steps, as well as evaluate zero-shot performance ($S = 0$). We consider an additional baseline of a randomly initialized model, which is trained from scratch on the finetuning samples.

Zero-shot and few-shot errors are given in Table 2. We find that cross-domain training achieves lower errors than other models during zero-shot and few-shot evaluation, even under an equal pretraining budget. A potential explanation for this is that training across domains prevents overfitting to a specific geometry/simulation setup and learns more general representations that can be used when finetuning to new data. We observe this in Figure 2, where zero-shot predictions a SHIFT-CCA geometry are plotted. When queried on unseen geometries, expert models tend to inherit biases from their pretraining data, for example predicting car-like features from DrivAerNet/ML or flow separation from Emmi Wing/Superwing that are not present in the target geometry. Appendix D provides additional visualizations, which further show the advantage of the joint model. Beyond improving generalization, cross-domain pretraining offers a practical benefit: existing datasets can be pooled to train a single model that serves all downstream tasks, eliminating the need to maintain a library of experts and select among them at deployment.

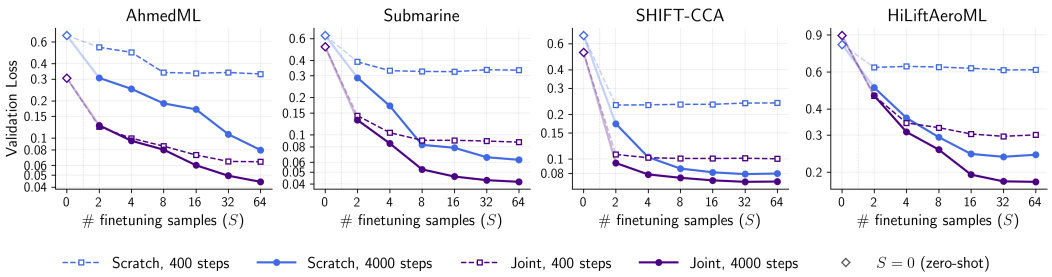

Figure 3: Validation loss vs. number of finetuning samples, plotted for the pretrained joint model and a model initialized from scratch. Two finetuning budgets are used (400/4000 steps). Note both axes are log scale.

Another observation is that experts close to the finetuning set tend to perform better and farther ones tend to perform worse. For example, adapting WindsorML to AhmedML performs well as both are simple automotive bodies, but adapting WindsorML to HiLiftAeroML performs poorly due to transferring from automotive to aerospace datasets (Table 2). In addition, the worst expert is always better than training from scratch in the few-shot setting, suggesting that any form of pretraining is beneficial in limited-data scenarios. However, in the zero-shot setting, the worst expert can be worse than a random initialization, so expert models can be unreliable when no finetuning data is available.

Sample Efficiency. To quantify sample efficiency, we compare the joint model to a fixed reference, chosen as the randomly initialized model. On the four held-out sets, we plot the validation loss as a function of the finetuning sample size S in Figure 3. At the original finetuning budget (400 steps), no randomly initialized model is able to outperform a joint model with $S = 2$, even as we increase S to the full held-out dataset size ($S > 64$). In this compute-limited setting, a joint model with only two samples outperforms a randomly initialized model that is provided the entire dataset.

This is important for fast adaptation of a pretrained model, however, we can often afford a larger finetuning budget, which we increase by 10 times (4000 steps). Given the error of the scratch model at a fixed sample size ($S = 64$) we compute what sample size the joint model needs to achieve the same error. Across the four finetuning sets, the joint model achieves the same error at $S \approx$ 4-8 samples, which is 8-16 times fewer samples. Furthermore, a pretrained model needs very little data to adapt: at any finetuning budget, just two training samples can cut the zero-shot error in half or more. With two finetuning samples, the NRMSE can also be competitive (~ 0.1), even though validation is performed on more than ten times as many samples (20-100), each with a new geometry. Given these performance gains, future CFD surrogates are likely to benefit substantially from a pretrained initialization. This benefit comes from using a pretrained model to accelerate convergence and achieve lower error on a new dataset, but also from reducing the number of samples that must be generated for a new application.

Different Architectures. To validate that the findings are agnostic to the underlying architecture, we replicate the primary pretraining and finetuning results for AB-UPT (Alkin et al., 2025a) and Transolver++ (Luo et al., 2025). Expert and joint models are pretrained and evaluated on the held-out datasets, and the results are shown in Appendix C.1. We find that the same trends hold for other architectures, where the joint model consistently outperforms any expert model during zero-shot and few-shot evaluation. This suggests that cross-domain pretraining is likely a good strategy for future architectures as well and complements ongoing work in CFD surrogates. We also verify that each architecture (SMART/AB-UPT/Transolver++) has similar pretraining errors (Appendix C.1).

4.2 Scaling Model and Dataset Size

If cross-dataset pretraining improves downstream performance, a natural question is how that performance scales with model size and dataset diversity. To test this, we train variants of the joint model on different numbers of pretraining datasets N_d as well as different model sizes N_p. We consider training on 2 datasets (DrivAerNet/Emmi Wing), 4 datasets (DrivAerNet/Emmi

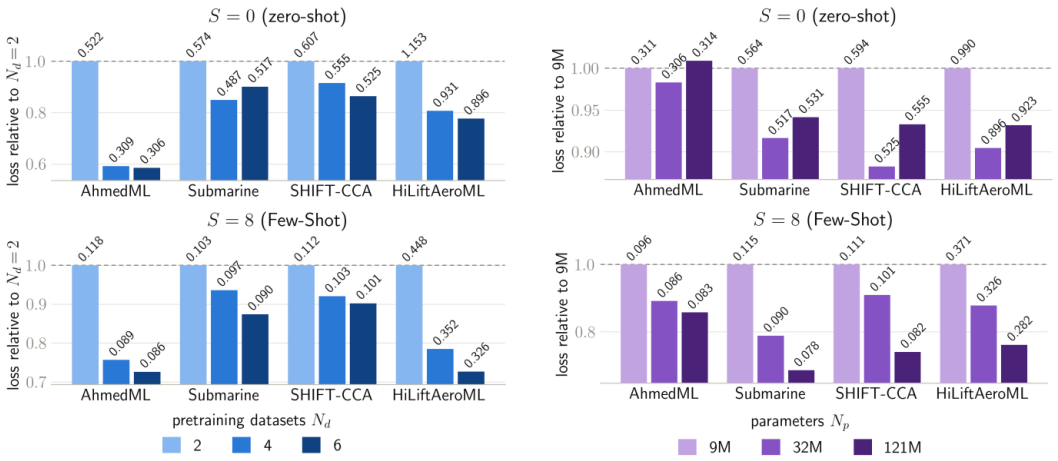

Figure 4: Error of joint models on held-out datasets, after using $S = 0$ or $S = 8$ finetuning samples. *Left:* Joint models are pretrained with different numbers of pretraining datasets (2/4/6), with the model size held constant ($N_p = 32$M). *Right:* Joint models are pretrained at varying model sizes (9/32/121M), with the size of the pretraining dataset held constant ($N_d = 6$).

Wing/WindsorML/Double Delta), as well as all 6 pretraining datasets. Furthermore, we consider 3 model sizes, with parameter counts of $N_p = 9/32/121$M. The zero-shot and few-shot performance of joint models on held-out datasets are shown in Figure 4. In general, we find that increasing the number of datasets seen during pretraining or the model size improves downstream performance. In the zero-shot setting there are some exceptions, where it seems that the largest model may overfit to the pretraining dataset. However, both large models and models trained on a diverse pretraining set are more sample efficient during finetuning, reaching lower errors at a fixed gradient and data budget (400 steps, $S = 8$).

The results suggest that the best way to achieve zero-shot generalization is by pretraining on a more diverse collection of data, rather than just increasing the model size. If data samples are available for the downstream application, then a larger model size can use these samples more efficiently. Therefore, generating a large and diverse pretraining dataset is the most effective lever for improving a cross-domain CFD surrogate, but it is also the main obstacle since simulation is likely more expensive than the model training itself. If this isn't feasible, a large model combined with a handful of samples ($S < 10$) per downstream application is also a viable strategy.

4.3 Pretraining Performance

The primary focus of this work examines model performance on new, unseen datasets. This has led to useful results and scaling trends; however, there are also some interesting findings from examining the pretraining performance. These may be relevant when models are deployed to standard, preexisting applications or just to understand what the tradeoffs are for cross-domain pretraining. We examine the performance of expert and joint models on the pretraining datasets in Table 3. For joint models, we also consider varying the number of pretraining datasets ($N_d = 2, 4, 6$) to include progressively more datasets.

For a fixed model size and compute budget, the best-performing model on each pretraining dataset is its corresponding expert, since that model sees only one dataset during training and can devote all of its parameters and gradient steps to fitting it. Training an expert is therefore the preferred strategy when a task has abundant data, though it sacrifices performance on the remaining datasets. We also find that experts generalize better to datasets similar to their own: the DrivAerNet model achieves lower errors on WindsorML and DrivAerML, whereas Double-Delta is the hardest target to generalize to, being the only full-body aircraft dataset.

Joint training trades low error on any single dataset for consistent performance across all of them. The trade-off is gradual: as the pretraining pool grows, per-dataset error rises slightly but average

Model	DrivAerNet	Emmi-Wing	WindsorML	Double-Delta	DrivAerML	SuperWing
DrivAerNet	0.143	0.855	0.470	0.973	0.401	0.830
Emmi-Wing	0.809	0.030	0.617	1.239	0.830	0.742
WindsorML	0.678	0.747	0.067	0.956	0.701	0.736
Double-Delta	0.766	0.657	0.587	0.057	0.781	0.668
DrivAerML	0.421	0.786	0.458	0.966	0.048	0.803
SuperWing	0.973	0.648	0.705	1.255	1.006	0.047
Joint ($N_d = 2$)	0.151	0.035	0.458	1.075	0.408	0.708
Joint ($N_d = 4$)	0.164	0.041	0.072	0.083	0.394	0.652
Joint ($N_d = 6$)	0.174	0.043	0.075	0.091	0.134	0.082

Table 3: Validation error of expert and joint models across the pretraining datasets. Joint models are trained at different pretraining dataset sizes $N_d = 2, 4, 6$. Cells are colored by error magnitude.

error improves, with no sign of catastrophic forgetting. Under a fixed parameter and gradient budget this is unavoidable since each added dataset reduces the steps and capacity available per task. Joint training is also a harder learning problem, requiring the model to accommodate multiple geometries and output fields, but we hypothesize that this difficulty is what yields better generalization. Scaling model size or pretraining compute largely recovers pretraining performance (Appendix C.4), but this can be misleading. Lower pretraining loss does not imply better transfer, as these scaled models can have higher zero-shot error on held-out domains, suggesting that overfitting can occur.

4.4 Examining Ambiguity in Cross-Domain Pretraining

Overview. Predicting a steady-state solution from geometry alone can be ill-defined, as the same geometry can map to different solutions depending on variables we cannot always observe. For example, different closure models, mesh resolutions, or modeling choices such as whether to use rotating wheels, can result in different solutions for the same geometry (Ashton et al., 2025b). Fully enumerating these in the simulation parameters θ is usually not feasible, however a domain-specific model can resolve this ambiguity by training on data with fixed simulation choices. However, in the cross-domain setting these hidden variables can vary across datasets, which we study in this section.

In the conventional, time-dependent PDE setting, this ambiguity is usually addressed by using context frames from the same rollout as a model input (McCabe et al., 2024). In this case, the context frames are guaranteed to have the same physics as the current state, which allows the model to infer the dynamics and uniquely forward the current state in time. In steady-state problems we do not have access to context frames; nevertheless, joint models are still able to resolve this ambiguity and distinguish different dynamics from the geometry G and simulation parameters θ alone. This is true even when the geometries originate from the same base geometry, such as in DrivAerNet/DrivAerML. Therefore, to study this behavior, we create toy problems where the geometry alone cannot determine the dynamics, as well as train joint models on the combined DrivAerNet/ML datasets.

Toy Problems. To study the ill-defined setting, we construct three toy problems in which the simulation parameters θ are withheld from the model. In the first (Double-Delta-α), a fixed set of geometries is solved across a range of angles of attack ($\alpha \in [11, 19]^\circ$), so that a single geometry maps to several distinct solutions (Figure 5). In the second (DrivAerNet-ν), DrivAerNet geometries are solved at two combinations of viscosity and inlet velocity, $(\nu, U) \in \{(1.0 \times 10^{-5},\ 20\,\mathrm{m/s}),\ (2.3 \times 10^{-5},\ 45\,\mathrm{m/s})\}$. The third mixes samples from DrivAerNet and DrivAerML that were selected for geometric similarity, with detailed underbodies and open wheels in both. For each problem we compare three variants that differ in what the model receives as input alongside the geometry: nothing (Vanilla), a paired simulation sharing the same θ (In-context), or θ itself (Oracle). In the DrivAerNet/ML case, the in-context sample is drawn from the matching dataset and the oracle signal is the dataset label. All models are the same size (32M) and are trained to the same budget (50k steps).

Results are reported in Table 4. For Double-Delta-α, we see that the vanilla model struggles without the parameter information, while the in-context model has a small penalty for needing to infer θ. This trend is also seen in DrivAerNet-ν, although it is a more difficult setting. Surprisingly, in the DrivAerNet/ML case, the vanilla model recovers performance identical to that of the oracle. While

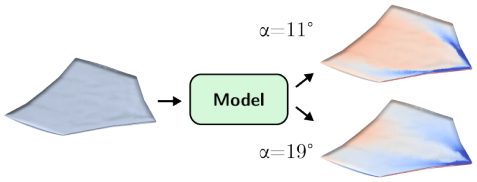

Figure 5: A case where the same geometry can have different physics.

Model Input:	G	G + context	$G + \theta$
Double-Delta-α	0.093	0.076	0.068
DrivAerNet-ν	0.450	0.164	0.150
DrivAerNet/ML	0.109	0.111	0.109

Table 4: Validation error on the toy problems, for models receiving the geometry G only (Vanilla), G plus a paired simulation sharing the same θ (In-Context), or G plus θ directly (Oracle).

the DrivAerNet/ML geometries are similar, it seems that differences in discretization or parametric morphing are enough for neural surrogates to distinguish the two. In this case, there is no benefit to providing context to the model during training and it can even be detrimental.

Takeaways. Our experiments show that cross-domain learning for CFD surrogates can succeed in practice by simply pooling datasets, without any context about the source domain or knowledge of hidden variables. In the DrivAerNet/ML case and when pretraining across six datasets, supplying an extra dataset label or context simulation does not help, and can even reduce performance slightly. We further discuss this in Appendix C.5.

This departure from context-based PDE foundation models reflects several differences in the steady-state CFD setting. First, CFD surrogates take a dense, unstructured point cloud as input, and unlike a fixed uniform grid, the discretization itself carries variation that the model can use to distinguish domains. Second, PDE benchmarks are typically built around well-defined splits over PDE terms or coefficients that can be inferred from context or supplied as conditioning (Takamoto et al., 2023; Gupta & Brandstetter, 2022). We construct two such splits as toy problems, but practical CFD settings are rarely so well-defined. For example, when training the oracle in DrivAerNet/ML, using the dataset index as conditioning is no longer physically meaningful, unlike for α, ν, U. Third, steady-state CFD datasets are constructed so that the output is determined by the geometry and simulation parameters alone, and therefore context or additional information is likely redundant. Additional information is needed for parameters that vary across domains (closure, solver, discretization), however these rarely vary within a geometry, since there is little incentive to solve the same geometry twice across separate datasets. Lastly, during finetuning, we assume a fixed simulation setup, so any remaining hidden variables are constant and can be absorbed few-shot. Therefore, in the current setting, cross-domain methods designed around PDE terms and coefficients appear unnecessary for CFD surrogates, unless a split has been deliberately constructed.

5 Conclusion

Overall, this work shows that cross-domain pretraining improves the zero- and few-shot performance of CFD surrogates on unseen datasets. A pretrained model yields lower error and reduces the number of samples that must be generated to deploy a surrogate in a new application. These gains also grow with both the diversity of the pretraining data and the size of the model.

While these results are encouraging, there a few limitations of this work. Firstly, the number of datasets considered is still small compared to cross-domain works in PDEs, and current CFD datasets likely have less diversity than time-dependent PDE benchmarks. Second, the main results assume ground truth normalization statistics for the finetuning sets, which may not always be available. However, this can be mostly addressed through non-dimensionalization, which we discuss in Appendix C.2. Lastly, we do not test the discretization invariance of the models, nor do we compute any errors on the full-resolution field (millions of cells). We rely on previous work that does this, where the full resolution generally has a slight increase in the error (for AB-UPT/SMART).

Despite these limitations, cross-domain pretraining shows clear promise as neural CFD surrogates expand into a wider range of engineering applications. We therefore advocate for continued work to generate steady-state CFD datasets spanning as many geometries and applications as possible. Pooled together, these datasets could support increasingly large, cross-domain models that serve as general-purpose CFD surrogates.

AI USE STATEMENT

In this work, we used generative AI tools for implementing methods, cleaning and reformatting datasets, and for supporting qualitative and thematic data analysis. We have not used generative AI tools to propose or refine hypotheses, design or provide feedback on research methodology or experiments, or interpret results. The rest of the tasks (generating synthetic data sets, developing theoretical models or conceptual frameworks, formulating mathematical claims, providing critical ingredients for proving mathematical claims, or assisting in the writing of proofs) are not applicable to this work.

We have reviewed all AI-assisted work. For coding agents, we have manually reviewed all implementations and verified that they work as intended. For data visualization and writing, we have taken raw outputs and modified them to our specific needs.

We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

ETHICS STATEMENT

There are no ethics considerations to disclose.

REPRODUCIBILITY STATEMENT

All source code and pretrained model checkpoints will be released to replicate results. Furthermore, all datasets will also be released in the processed, formatted versions, with the exception of datasets generated by private companies.

REFERENCES

Benedikt Alkin, Maurits Bleeker, Richard Kurle, Tobias Kronlachner, Reinhard Sonnleitner, Matthias Dorfer, and Johannes Brandstetter. Ab-upt: Scaling neural cfd surrogates for high-fidelity automotive aerodynamics simulations via anchored-branched universal physics transformers, 2025a. URL https://arxiv.org/abs/2502.09692.

Benedikt Alkin, Richard Kurle, Louis Serrano, Dennis Just, and Johannes Brandstetter. Ab-upt for automotive and aerospace applications, 2025b. URL https://arxiv.org/abs/2510.15808.

Neil Ashton, Danielle C. Maddix, Samuel Gundry, and Parisa M. Shabestari. Ahmedml: High-fidelity computational fluid dynamics dataset for incompressible, low-speed bluff body aerodynamics, 2024. URL https://arxiv.org/abs/2407.20801.

Neil Ashton, Jordan B. Angel, Aditya S. Ghate, Gaetan K. W. Kenway, Man Long Wong, Cetin Kiris, Astrid Walle, Danielle C. Maddix, and Gary Page. Windsorml: High-fidelity computational fluid dynamics dataset for automotive aerodynamics, 2025a. URL https://arxiv.org/abs/2407.19320.

Neil Ashton, Johannes Brandstetter, and Siddhartha Mishra. Fluid intelligence: A forward look on ai foundation models in computational fluid dynamics, 2025b. URL https://arxiv.org/abs/2511.20455.

Neil Ashton, Charles Mockett, Marian Fuchs, Louis Fliessbach, Hendrik Hetmann, Thilo Knacke, Norbert Schonwald, Vangelis Skaperdas, Grigoris Fotiadis, Astrid Walle, Burkhard Hupertz, and Danielle Maddix. Drivaerml: High-fidelity computational fluid dynamics dataset for road-car external aerodynamics, 2025c. URL https://arxiv.org/abs/2408.11969.

Neil Ashton, Adam Clark, Liam Heidt, Christopher Ivey, Sanjeeb Bose, Rahul Agrawal, Konrad Goc, Rishi Ranade, Corey Adams, Peter Sharpe, Sheel Nidhan, Semit Akkurt, Daniel Leibovici, and Jean Kossaifi. Hiliftaeroml: High-fidelity computational fluid dynamics dataset for high-lift aircraft aerodynamics, 2026. URL https://arxiv.org/abs/2605.19565.

Mahmoud Assran, Quentin Duval, Ishan Misra, Piotr Bojanowski, Pascal Vincent, Michael Rabbat, Yann LeCun, and Nicolas Ballas. Self-supervised learning from images with a joint-embedding predictive architecture, 2023. URL https://arxiv.org/abs/2301.08243.

Matthieu Blanke and Marc Lelarge. Interpretable meta-learning of physical systems, 2024. URL https://arxiv.org/abs/2312.00477.

Wuyang Chen, Jialin Song, Pu Ren, Shashank Subramanian, Dmitriy Morozov, and Michael W. Mahoney. Data-efficient operator learning via unsupervised pretraining and in-context learning, 2025. URL https://arxiv.org/abs/2402.15734.

Pedro M. P. Curvo, Jan-Willem van de Meent, and Maksim Zhdanov. Mspt: Efficient large-scale physical modeling via parallelized multi-scale attention, 2026. URL https://arxiv.org/abs/2512.01738.

Mohamed Elrefaie, Florin Morar, Angela Dai, and Faez Ahmed. Drivaernet++: A large-scale multi-modal car dataset with computational fluid dynamics simulations and deep learning benchmarks, 2025. URL https://arxiv.org/abs/2406.09624.

Francisco Giral, Abhijeet Vishwasrao, Andrea Arroyo Ramo, Mahmoud Golestanian, Federica Tonti, Adrian Lozano-Duran, Steven L. Brunton, Sergio Hoyas, Hector Gomez, Soledad Le Clainche, and Ricardo Vinuesa. Aerojepa: Learning semantic latent representations for scalable 3d aerodynamic field modeling, 2026. URL https://arxiv.org/abs/2605.05586.

Jayesh K. Gupta and Johannes Brandstetter. Towards multi-spatiotemporal-scale generalized pde modeling, 2022. URL https://arxiv.org/abs/2209.15616.

Jan Hagnberger and Mathias Niepert. Smart: Scalable mesh-free aerodynamic simulations from raw geometries using a transformer-based surrogate model, 2026. URL https://arxiv.org/abs/2601.18707.

Jan Hagnberger, Daniel Musekamp, and Mathias Niepert. Calm-pde: Continuous and adaptive convolutions for latent space modeling of time-dependent pdes, 2025. URL https://arxiv.org/abs/2505.12944.

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, 2023. URL https://arxiv.org/abs/2302.14376.

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, 2024. URL https://arxiv.org/abs/2403.03542.

Kaiming He, Xinlei Chen, Saining Xie, Yanghao Li, Piotr Dollár, and Ross Girshick. Masked autoencoders are scalable vision learners, 2021. URL https://arxiv.org/abs/2111.06377.

Maximilian Herde, Bogdan Raonić, Tobias Rohner, Roger Käppeli, Roberto Molinaro, Emmanuel de Bézenac, and Siddhartha Mishra. Poseidon: Efficient foundation models for pdes, 2024. URL https://arxiv.org/abs/2405.19101.

Anran Jiao, Haiyang He, Rishikesh Ranade, Jay Pathak, and Lu Lu. One-shot learning for solution operators of partial differential equations. *Nature Communications*, 16(1): 8386, 2025. doi: 10.1038/s41467-025-63076-z. URL https://doi.org/10.1038/s41467-025-63076-z.

Seunghwan Keum and Alok Warey. Adapting automotive aerodynamics surrogates to new vehicle families via transfer learning, 2026. URL https://arxiv.org/abs/2605.27968.

Matthieu Kirchmeyer, Yuan Yin, Jérémie Donà, Nicolas Baskiotis, Alain Rakotomamonjy, and Patrick Gallinari. Generalizing to new physical systems via context-informed dynamics model, 2022. URL https://arxiv.org/abs/2202.01889.

Felix Koehler, Simon Niedermayr, Rüdiger Westermann, and Nils Thuerey. Apebench: A benchmark for autoregressive neural emulators of pdes. In *Advances in Neural Information Processing Systems 37*, NeurIPS 2024, pp. 120252–120310. Neural Information Processing Systems Foundation, Inc. (NeurIPS), 2024. doi: 10.52202/079017-3822. URL http://dx.doi.org/10.52202/079017-3822.

Armand Kassaï Koupaï, Jorge Mifsut Benet, Yuan Yin, Jean-Noël Vittaut, and Patrick Gallinari. Geps: Boosting generalization in parametric pde neural solvers through adaptive conditioning, 2024. URL https://arxiv.org/abs/2410.23889.

Armand Kassaï Koupaï, Lise Le Boudec, and Patrick Gallinari. Efficient generative transformer operators for million-point pdes, 2025. URL https://arxiv.org/abs/2512.04974.

Remi Lam, Alvaro Sanchez-Gonzalez, Matthew Willson, Peter Wirnsberger, Meire Fortunato, Ferran Alet, Suman Ravuri, Timo Ewalds, Zach Eaton-Rosen, Weihua Hu, Alexander Merose, Stephan Hoyer, George Holland, Oriol Vinyals, Jacklynn Stott, Alexander Pritzel, Shakir Mohamed, and Peter Battaglia. Graphcast: Learning skillful medium-range global weather forecasting, 2023. URL https://arxiv.org/abs/2212.12794.

Xinyi Li, Zongyi Li, Nikola Kovachki, and Anima Anandkumar. Geometric operator learning with optimal transport, 2025a. URL https://arxiv.org/abs/2507.20065.

Zijie Li, Kazem Meidani, and Amir Barati Farimani. Transformer for partial differential equations' operator learning, 2023a. URL https://arxiv.org/abs/2205.13671.

Zijie Li, Anthony Zhou, and Amir Barati Farimani. Generative latent neural pde solver using flow matching, 2025b. URL https://arxiv.org/abs/2503.22600.

Zongyi Li, Daniel Zhengyu Huang, Burigede Liu, and Anima Anandkumar. Fourier neural operator with learned deformations for pdes on general geometries. *J. Mach. Learn. Res.*, 24(1), January 2023b. ISSN 1532-4435.

Zongyi Li, Nikola Borislavov Kovachki, Chris Choy, Boyi Li, Jean Kossaifi, Shourya Prakash Otta, Mohammad Amin Nabian, Maximilian Stadler, Christian Hundt, Kamyar Azizzadenesheli, and Anima Anandkumar. Geometry-informed neural operator for large-scale 3d pdes, 2023c. URL https://arxiv.org/abs/2309.00583.

Mario Lino, Tobias Pfaff, and Nils Thuerey. Learning distributions of complex fluid simulations with diffusion graph networks, 2025. URL https://arxiv.org/abs/2504.02843.

Qiang Liu, Felix Koehler, Benjamin Holzschuh, and Nils Thuerey. Tadpole: Autoencoders as foundation models for 3d pdes with online learning, 2026. URL https://arxiv.org/abs/2605.15284.

Luminary Cloud. Shift-cca: High-fidelity computational fluid dynamics dataset for collaborative combat aircraft aerodynamics, 2025a. URL https://huggingface.co/datasets/luminary-shift/CCA-sample.

Luminary Cloud. Shift-submarine: High-fidelity computational fluid dynamics dataset for submarine hydrodynamics, 2025b. URL https://huggingface.co/datasets/luminary-shift/Submarine-sample.

Huakun Luo, Haixu Wu, Hang Zhou, Lanxiang Xing, Yichen Di, Jianmin Wang, and Mingsheng Long. Transolver++: An accurate neural solver for pdes on million-scale geometries, 2025. URL https://arxiv.org/abs/2502.02414.

Siddharth Mansingh, James Amarel, Ragib Arnab, Arvind Mohan, Kamaljeet Singh, Gerd J. Kunde, Nicolas Hengartner, Benjamin Migliori, Emily Casleton, Nathan A. Debardeleben, Ayan Biswas, Diane Oyen, and Earl Lawrence. Towards reasoning for pde foundation models: A reward-model-driven inference-time-scaling algorithm, 2026. URL https://arxiv.org/abs/2509.02846.

Michael McCabe, Bruno Régaldo-Saint Blancard, Liam Holden Parker, Ruben Ohana, Miles Cranmer, Alberto Bietti, Michael Eickenberg, Siavash Golkar, Geraud Krawezik, Francois Lanusse, Mariel Pettee, Tiberiu Tesileanu, Kyunghyun Cho, and Shirley Ho. Multiple physics pretraining for physical surrogate models, 2024. URL https://arxiv.org/abs/2310.02994.

Michael McCabe, Payel Mukhopadhyay, Tanya Marwah, Bruno Regaldo-Saint Blancard, Francois Rozet, Cristiana Diaconu, Lucas Meyer, Kaze W. K. Wong, Hadi Sotoudeh, Alberto Bietti, Irina Espejo, Rio Fear, Siavash Golkar, Tom Hehir, Keiya Hirashima, Geraud Krawezik, Francois Lanusse, Rudy Morel, Ruben Ohana, Liam Parker, Mariel Pettee, Jeff Shen, Kyunghyun Cho, Miles Cranmer, and Shirley Ho. Walrus: A cross-domain foundation model for continuum dynamics, 2026. URL https://arxiv.org/abs/2511.15684.

Nick McGreivy and Ammar Hakim. Weak baselines and reporting biases lead to overoptimism in machine learning for fluid-related partial differential equations. *Nature Machine Intelligence*, 6(10):1256–1269, 2024. ISSN 2522-5839. doi: 10.1038/s42256-024-00897-5. URL http://dx.doi.org/10.1038/s42256-024-00897-5.

Rudy Morel, Jiequn Han, and Edouard Oyallon. Disco: learning to discover an evolution operator for multi-physics-agnostic prediction, 2025. URL https://arxiv.org/abs/2504.19496.

Payel Mukhopadhyay, Stefan S. Nixon, Romain Watteaux, Michael McCabe, Alberto Bietti, Kyunghyun Cho, Cristiana Diaconu, Irina Espejo Morales, David Fouhey, Siavash Golkar, Tom Hehir, Shirley Ho, Jake Kovalic, Geraud Krawezik, Francois Lanusse, Tanya Marwah, Rudy Morel, Mariel Pettee, Helen Qu, Jeff Shen, Hadi Sotoudeh, Stuart B. Dalziel, and Miles Cranmer. Emergent transfer of a physics foundation model from simulation to laboratory turbulence, 2026. URL https://arxiv.org/abs/2606.01470.

Ruben Ohana, Michael McCabe, Lucas Meyer, Rudy Morel, Fruzsina J. Agocs, Miguel Beneitez, Marsha Berger, Blakesley Burkhart, Keaton Burns, Stuart B. Dalziel, Drummond B. Fielding, Daniel Fortunato, Jared A. Goldberg, Keiya Hirashima, Yan-Fei Jiang, Rich R. Kerswell, Suryanarayana Maddu, Jonah Miller, Payel Mukhopadhyay, Stefan S. Nixon, Jeff Shen, Romain Watteaux, Bruno Régaldo-Saint Blancard, François Rozet, Liam H. Parker, Miles Cranmer, and Shirley Ho. The well: a large-scale collection of diverse physics simulations for machine learning, 2025. URL https://arxiv.org/abs/2412.00568.

Fabian Paischer, Leo Cotteleer, Yann Dreze, Richard Kurle, Dylan Rubini, Maurits Bleeker, Tobias Kronlachner, and Johannes Brandstetter. Going with the speed of sound: Pushing neural surrogates into highly-turbulent transonic regimes, 2026. URL https://arxiv.org/abs/2511.21474.

Ethan Perez, Florian Strub, Harm de Vries, Vincent Dumoulin, and Aaron Courville. Film: Visual reasoning with a general conditioning layer, 2017. URL https://arxiv.org/abs/1709.07871.

Tobias Pfaff, Meire Fortunato, Alvaro Sanchez-Gonzalez, and Peter W. Battaglia. Learning mesh-based simulation with graph networks, 2021. URL https://arxiv.org/abs/2010.03409.

Helen Qu, Rudy Morel, Michael McCabe, Alberto Bietti, François Lanusse, Shirley Ho, and Yann LeCun. Representation learning for spatiotemporal physical systems, 2026. URL https://arxiv.org/abs/2603.13227.

Rishikesh Ranade, Mohammad Amin Nabian, Kaustubh Tangsali, Alexey Kamenev, Oliver Hennigh, Ram Cherukuri, and Sanjay Choudhry. Domino: A decomposable multi-scale iterative neural operator for modeling large scale engineering simulations, 2025. URL https://arxiv.org/abs/2501.13350.

Stephan Rasp, Stephan Hoyer, Alexander Merose, Ian Langmore, Peter Battaglia, Tyler Russel, Alvaro Sanchez-Gonzalez, Vivian Yang, Rob Carver, Shreya Agrawal, Matthew Chantry, Zied Ben Bouallegue, Peter Dueben, Carla Bromberg, Jared Sisk, Luke Barrington, Aaron Bell, and Fei Sha. Weatherbench 2: A benchmark for the next generation of data-driven global weather models, 2024. URL https://arxiv.org/abs/2308.15560.

Mahindra Singh Rautela, Alexander Most, Siddharth Mansingh, Bradley C. Love, Alexander Scheinker, Diane Oyen, Nathan Debardeleben, Earl Lawrence, and Ayan Biswas. Morph: Pde foundation models with arbitrary data modality, 2026. URL https://arxiv.org/abs/2509.21670.

Louis Serrano, Armand Kassaï Koupaï, Thomas X Wang, Pierre Erbacher, and Patrick Gallinari. Zebra: In-context generative pretraining for solving parametric pdes, 2025. URL https://arxiv.org/abs/2410.03437.

Louis Serrano, Jiequn Han, Edouard Oyallon, Shirley Ho, and Rudy Morel. Test-time generalization for physics through neural operator splitting. *arXiv preprint arXiv:2602.00884*, 2026.

Noam Shazeer. Glu variants improve transformer, 2020. URL https://arxiv.org/abs/2002.05202.

Yiren Shen and Juan J. Alonso. A multi-fidelity double-delta wing dataset and empirical scaling laws for gnn-based aerodynamic field surrogates. In *AIAA SCITECH 2026 Forum*. American Institute of Aeronautics and Astronautics, January 2026. doi: 10.2514/6.2026-0686. URL http://dx.doi.org/10.2514/6.2026-0686.

Shashank Subramanian, Peter Harrington, Kurt Keutzer, Wahid Bhimji, Dmitriy Morozov, Michael Mahoney, and Amir Gholami. Towards foundation models for scientific machine learning: Characterizing scaling and transfer behavior, 2023. URL https://arxiv.org/abs/2306.00258.

Jingmin Sun, Yuxuan Liu, Zecheng Zhang, and Hayden Schaeffer. Towards a foundation model for partial differential equations: Multi-operator learning and extrapolation, 2025. URL https://arxiv.org/abs/2404.12355.

Makoto Takamoto, Francesco Alesiani, and Mathias Niepert. Learning neural PDE solvers with parameter-guided channel attention. 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. 33448–33467. PMLR, 23–29 Jul 2023. URL https://proceedings.mlr.press/v202/takamoto23a.html.

Makoto Takamoto, Timothy Praditia, Raphael Leiteritz, Dan MacKinlay, Francesco Alesiani, Dirk Pflüger, and Mathias Niepert. Pdebench: An extensive benchmark for scientific machine learning, 2024. URL https://arxiv.org/abs/2210.07182.

Nicholas Thumiger, Andrea Bartezzaghi, Mattia Rigotti, Cezary Skura, Thomas Frick, Elisa Serioli, Fabrizio Arbucci, and A. Cristiano I. Malossi. Faster by design: Interactive aerodynamics via neural surrogates trained on expert-validated cfd, 2026. URL https://arxiv.org/abs/2604.18491.

Tian Wang and Chuang Wang. Latent neural operator for solving forward and inverse pde problems, 2024. URL https://arxiv.org/abs/2406.03923.

Haixu Wu, Huakun Luo, Haowen Wang, Jianmin Wang, and Mingsheng Long. Transolver: A fast transformer solver for pdes on general geometries, 2024. URL https://arxiv.org/abs/2402.02366.

Liu Yang, Siting Liu, Tingwei Meng, and Stanley J. Osher. In-context operator learning with data prompts for differential equation problems. *Proceedings of the National Academy of Sciences*, 120(39), 2023. ISSN 1091-6490. doi: 10.1073/pnas.2310142120. URL http://dx.doi.org/10.1073/pnas.2310142120.

Yunjia Yang, Weishao Tang, Mengxin Liu, Nils Thuerey, Yufei Zhang, and Haixin Chen. Superwing: a comprehensive transonic wing dataset for data-driven aerodynamic design, 2026. URL https://arxiv.org/abs/2512.14397.

Zhanhong Ye, Zining Liu, Bingyang Wu, Hongjie Jiang, Leheng Chen, Minyan Zhang, Xiang Huang, Qinghe Meng. Jingyuan Zou, Hongsheng Liu, and Bin Dong. Pdeformer-2: A versatile foundation model for two-dimensional partial differential equations, 2026. URL https://arxiv.org/abs/2507.15409.

Maksim Zhdanov, Max Welling, and Jan-Willem van de Meent. Erwin: A tree-based hierarchical transformer for large-scale physical systems, 2025. URL https://arxiv.org/abs/2502.17019.

Maksim Zhdanov, Ana Lucic, Max Welling, and Jan-Willem van de Meent. (sparse) attention to the details: Preserving spectral fidelity in ml-based weather forecasting models, 2026. URL https://arxiv.org/abs/2604.16429.

Anthony Zhou and Amir Barati Farimani. Masked autoencoders are pde learners, 2024. URL https://arxiv.org/abs/2403.17728.

Anthony Zhou, Cooper Lorsung, AmirPouya Hemmasian, and Amir Barati Farimani. Strategies for pretraining neural operators, 2024. URL https://arxiv.org/abs/2406.08473.

Anthony Zhou, Zijie Li, Michael Schneier, John R Buchanan Jr, and Amir Barati Farimani. Text2pde: Latent diffusion models for accessible physics simulation, 2025. URL https://arxiv.org/abs/2410.01153.

Hang Zhou, Haixu Wu, Haonan Shangguan, Yuezhou Ma, Huikun Weng, Jianmin Wang, and Mingsheng Long. Transolver-3: Scaling up transformer solvers to industrial-scale geometries, 2026. URL https://arxiv.org/abs/2602.04940.

A Dataset Details

Overview. We give additional details on each dataset, such as how the original data was generated, the relevant parameters, and any post-processing done for this work. Additionally, to visualize each dataset we plot the geometry, coefficient of pressure C_p, and x-component of the skin friction coefficient C_{f_x} for a single sample in Figure 6. Furthermore, we plot the corresponding volumetric fields (coefficient of pressure C_p and non-dimensional velocity magnitude $|u^*| = |\vec{u}|/U_\infty$) for these samples in Figure 7.

For all datasets, we crop the volume domain to exclude far-field regions where the velocity and pressure fields are effectively constant. During preprocessing, each sample is loaded from its raw format (e.g., .stl, .vtp, .vtu) and converted into a NumPy memory-mapped array (.npy). Before writing to disk, we randomly permute the points within each sample so that any contiguous slice of the memory map yields a random subset of points and their associated field values. This lets us draw random point subsets by reading contiguous slices, rather than loading entire samples (which can contain millions of points) into memory, and enables fast data loading.

Finally, the datasets can vary widely in their spatial discretization. High-resolution, scale-resolving simulations typically use local mesh refinement in regions of strong gradients, and some datasets fully resolve the boundary layer, which concentrates a large fraction of the volume mesh near the geometry. As a result, randomly sampling a batch of volume points can under-represent features in the freestream. To mitigate this and make the effective discretization more consistent across datasets, we adopt a sampling scheme that produces a more spatially uniform point distribution. This is a design choice that we ablate in Appendix C.3. To sample N points, we first draw an oversampled set of kN points from storage, then select N points from this set with probability inversely proportional to the occupancy of their cell on a voxel grid. A final note is that normalization statistics are computed under this measure.

A.1 DrivAerNet++ (Elrefaie et al., 2025)

Overview. The dataset consists of 8000 parametrically morphed DrivAer passenger-car bodies. These consist of different base geometries (fastback, estateback and notchback), with open/closed and smooth/detailed wheels, and detailed or smooth underbodies, and are morphed with 26 geometric design parameters.

Data Generation. Samples are solved with a steady, incompressible RANS solver with a $\kappa - \omega$ SST closure, using OpenFOAM v11, SIMPLE coupling, and a wall-function (nutUSpaldingWallFunction) for the boundary layer. There are around 24 M cells per case (500–750 k on the car surface). 7000 iterations are solved per case, with forces averaged over the last 1000.

Parameters. U_∞ = 30 m/s, Re 8.37e6–1.01e7 (varying with car length), air at ν = 1.56e-5 m^2/s, ρ = 1.184 kg/m^3.

Processing. Certain samples assume a symmetric boundary condition and therefore only have +y values. To be consistent with DrivAerML, we mirror the geometry and fields to resemble a full car. Furthermore, we negate OpenFOAM's shear stress convention, such that positive wall-shear stresses point in the direction of flow. The volume domain is cropped based on the geometry bounding box; given a bounding box $[(x_{min}, y_{min}, z_{min}), (x_{max}, y_{max}, z_{max})]$ of size $L = (L_x, L_y, L_z)$, the volume domain is cropped to $[x_{min} - 0.5L_x, x_{max} + 2.0L_x]$ (wake is +x), $[y_{min} - 0.75L_y, y_{max} + 0.75L_y]$ (symmetric) and $[z_{min}, z_{max} + 0.5L_z]$ (no downward extension due to the floor).

A.2 DrivAerML (Ashton et al., 2025c)

Overview. The dataset consists of 500 morphs of the DrivAer notchback body, released as time-averaged hybrid RANS/LES statistics rather than as a single converged solution.

Data Generation. Samples are solved with an SA-based σ-DDES: a Spalart-Allmaras RANS branch with the Nicoud σ-model as the subgrid-scale operator, plus Enhanced Protection shielding to keep LES content out of attached boundary layers, and Spalding all-y^+ wall functions. The

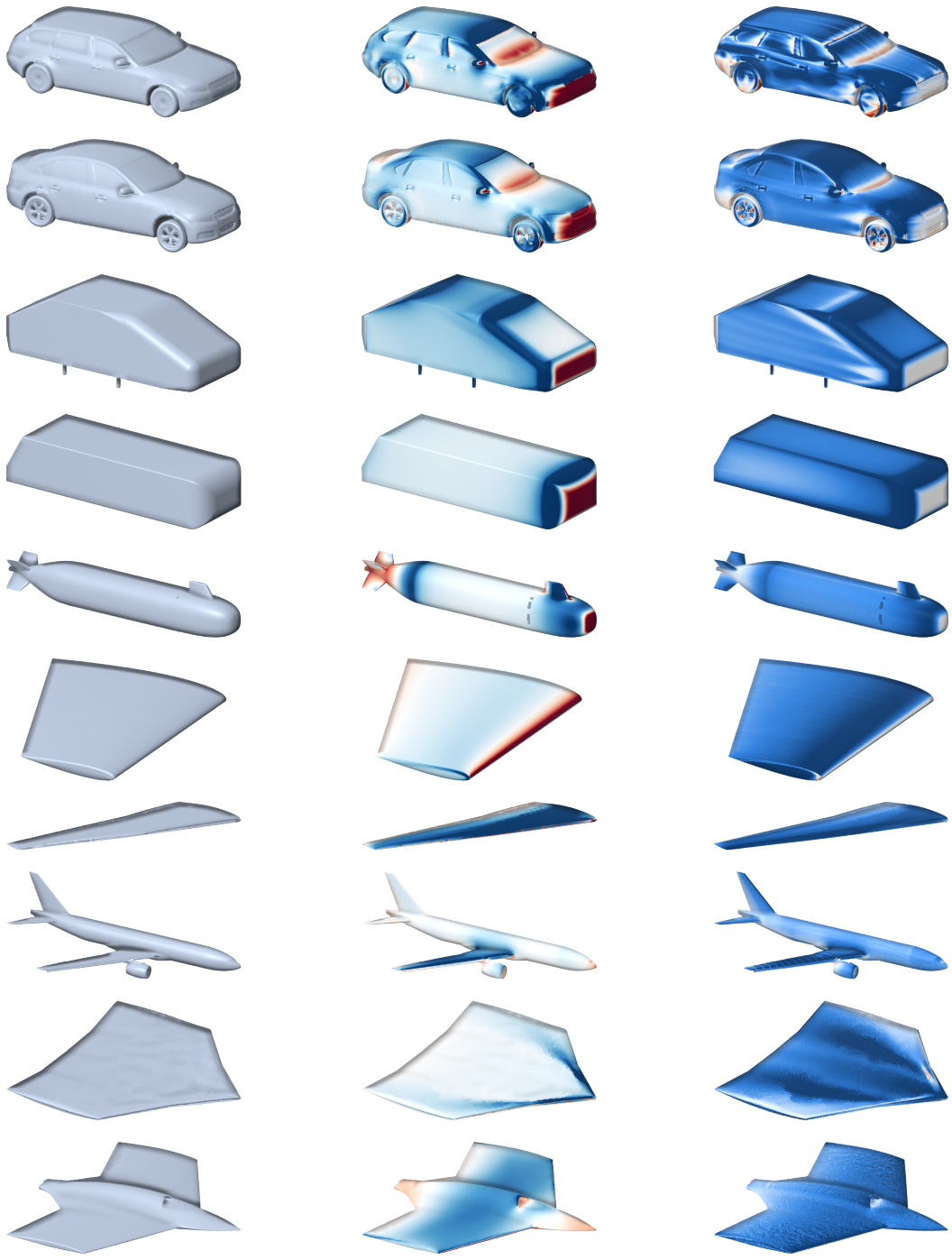

Figure 6: The geometry (left), coefficient of pressure C_p (center), and x-component of the skin friction coefficient C_{f_x} (right) visualized for a single sample in all datasets. From top to bottom: DrivAerNet, DrivAerML, WindsorML, AhmedML, SHIFT-Submarine, Emmi Wing, SuperWing, HiLiftAeroML, Double Delta, SHIFT-CCA.

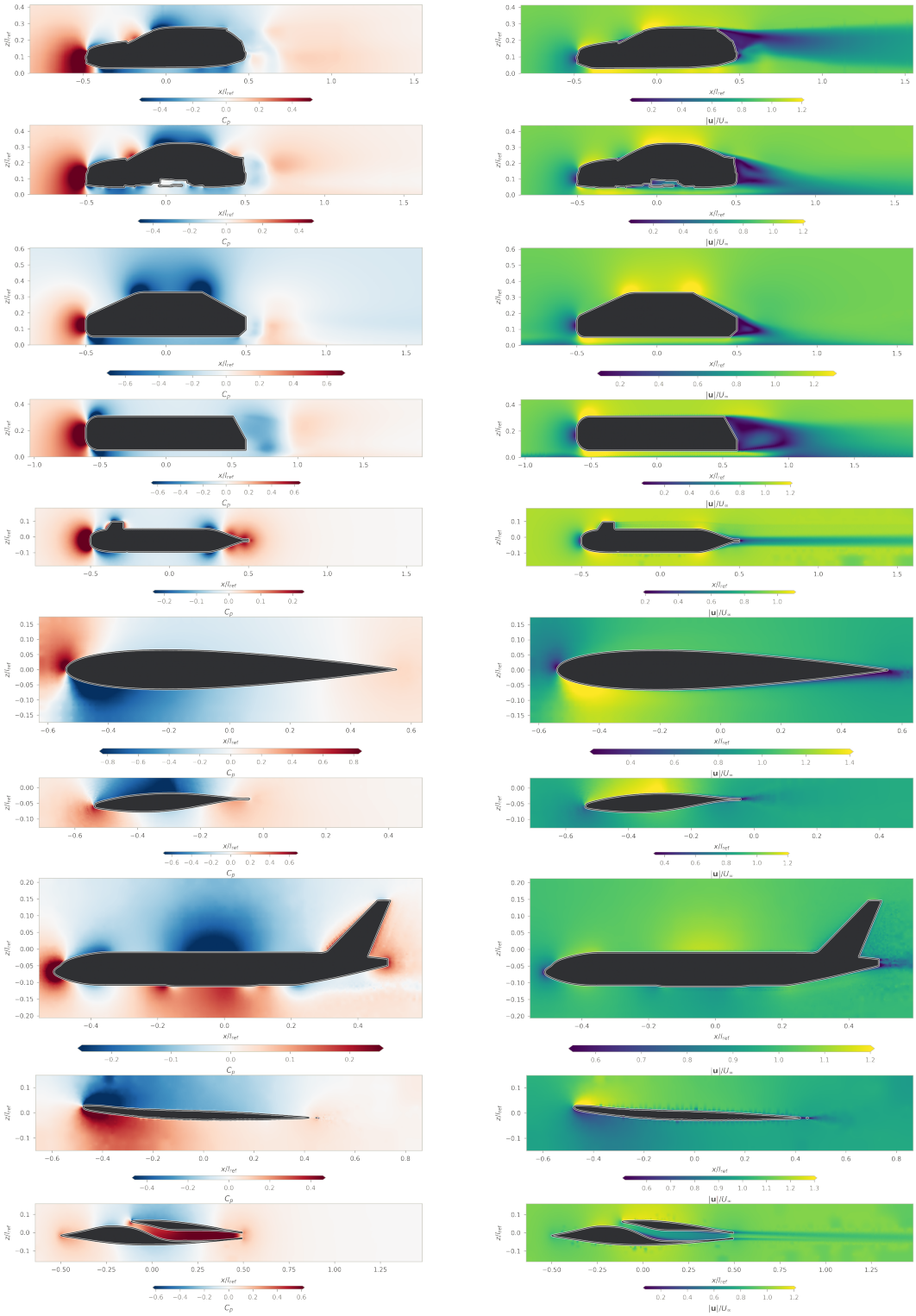

Figure 7: The volumetric pressure (shown as the coefficient of pressure C_p) and non-dimensional volumetric velocity magnitude $|\vec{u}|/U_\infty$, plotted for a single sample in all datasets. From top to bottom: DrivAerNet, DrivAerML, WindsorML, AhmedML, SHIFT-Submarine, Emmi Wing, SuperWing, HiLiftAeroML, Double Delta, SHIFT-CCA.

solver is a custom `pimpleFoam` derivative (OpenFOAM v2212) with a modified Rhie-Chow interpolation and second-order implicit Euler time integration. Meshes carry around 160 M cells, with a first boundary-layer height of 0.75 mm and seven layers to 12 mm. Statistics are accumulated until the drag estimate settles to ± 1.5 drag counts, generally 40–60 convective time units.

Parameters. U_∞ = 38.889 m/s, Re = 7.2e6 based on the 2.786 m wheelbase (1.2e7 if length-based), air at ν = 1.507e-5 m^2/s, ρ = 1.2041 kg/m^3.

Processing. The ground plane sits at $z \approx -0.318$ rather than $z = 0$, but subtracting the geometry centroid during normalization accounts for this. The volume field is cropped identically to DrivAerNet samples. OpenFOAM's shear stress sign convention is also flipped such that positive wall-shear stresses point in the direction of flow.

A.3 AHMEDML (ASHTON ET AL., 2024)

Overview. The dataset consists of 500 variants of the Ahmed body, the canonical simplified automotive bluff body, spanning slant angle, body length, width and height, and ground clearance.

Data Generation. Samples are solved with a plain SA-DDES using the standard f_d shielding function, similar to DrivAerML, but without the σ-model and Enhanced Protection. Meshes carry 15–21 M prismatic and hex-dominant cells and are coarse at the wall; there are three prism layers yielding $y^+ \approx 50$ and the boundary layer is modeled with Spalding wall functions. Each case runs for 40 convective transit times with time-averaging beginning after the first 10.

Parameters. U_∞ = 1 m/s, with Re = 7.68e5 based on the body height (2.78e6 if length-based). The density and viscosity are set at ρ = 1 kg/m^3 and ν = 3.75e-7 m^2/s.

Processing. The tunnel floor is at $z = 0$, but the Ahmed body is offset on stilts so the lowest geometry point sits at $z \approx 0.05$; the volume crop therefore clamps its lower bound down to the ground plane rather than to the surface bounding box. Otherwise the volume crop is identical to that of DrivAerNet/DrivAerML. OpenFOAM's shear stress sign convention is flipped as well.

A.4 WINDSORML (ASHTON ET AL., 2025A)

Overview. The dataset consists of 355 variants of the Windsor body, a squareback automotive reference geometry, released as time-averaged wall-modelled LES. It is the dataset with no RANS anywhere in the closure.

Data Generation. Samples are solved with wall-modelled LES using the constant-coefficient Vreman subgrid-scale model and an equilibrium wall model imposed as a shear stress constraint. The solver (Volcano ScaLES) is compressible and explicit, nominally fourth-order in space with SSP-RK3 time stepping, on Cartesian octree grids with an immersed-boundary geometry representation. The baseline grid carries 275 M cells at 0.75 mm minimum spacing, and fields are time-averaged.

Parameters. U_∞ = 40 m/s, Re = 2.9e6 based on the 1.044 m body length. The density and viscosity are set to ρ = 1.2712 kg/m^3 and ν = 1.44e-5 m^2/s.

Processing. The native dataset axes differ from other datasets: x is streamwise but y is vertical, with the floor at $y \approx 0$, and z is lateral. Therefore we apply a signed permutation $(x, y, z) \mapsto (x, -z, y)$ to positions, the surface shear stress and to the volume velocity, which brings the dataset into the shared frame. An identical volume crop to the prior automotive datasets is applied. The dataset did not ship with a freestream pressure, so we compute one at p_∞ = 37340.5 Pa to compute non-dimensional values.

A.5 SHIFT-SUBMARINE (LUMINARY CLOUD, 2025B)

Overview. The dataset consists of 101 parameterized Myring-type submarine hulls, varied over more than twenty shape parameters covering cross-section, fore- and aft-body profiles, sail and fin placement, and length-to-diameter ratio. It is the only dataset that is in water.

Data Generation. Specific details are not disclosed, but we assume that samples are solved as steady RANS with a Spalart-Allmaras closure on the Luminary finite-volume solver. Meshes carry around 5.6 M volume points and 32 M tetrahedra, with about 0.8 M surface points per case.

Parameters. U_∞ = 5 m/s in water, ρ = 998 kg/m^3, $\nu \approx$ 1.0e-6 m^2/s, hull length 6.455 m, so Re $\approx$ 3.2e7.

Processing. The raw wall-shear stress seems to have numerical artifacts at the strake and sail edges, with $\max|C_f| \approx 0.10$ against a 99.9th percentile of 0.013, so we drop points beyond $\pm 3\sigma$ on any shear component, computed per sample. The volume crop is isotropic and symmetric in the y-z plane, with no floor treatment, since the body is fully submerged. Given a bounding box $[(x_{min}, y_{min}, z_{min}), (x_{max}, y_{max}, z_{max})]$ of size $L = (L_x, L_y, L_z)$, the volume domain is cropped to $[x_{min} - 0.5L_x, x_{max} + 2.0L_x]$, $[y_{min} - 0.75L_y, y_{max} + 0.75L_y]$, and $[z_{min} - 0.75L_z, z_{max} + 0.75L_z]$.

A.6 EMMI-WING (PAISCHER ET AL., 2026)

Overview. The dataset consists of 29,609 steady compressible solutions over parameterized tapered swept wings, described by four geometric parameters (root chord, span, taper ratio and sweep) and two inflow parameters.

Data Generation. Samples are solved with OpenFOAM v2506 `rhoSimpleFoam` assuming a steady, compressible, perfect gas, with a Spalart-Allmaras closure and second-order schemes. Body-fitted `snappyHexMesh` grids carry prismatic boundary layers at $y^+ \in [50, 200]$ and the boundary layer is modeled rather than resolved. Meshes hold around 3.3 M volume points and 46–114 k surface points.

Parameters. The freestream state is computed to be: p_∞ = 100000 Pa, ρ_∞ = 1.16639 kg/m^3 and T_∞ = 298 K with R = 287.7 $J/(kg\,K)$. U_∞ varies, over 150–300 m/s, giving Mach 0.437–0.875 and Re 5–20e6 at angles of attack of $-10°$ to $10°$. Angle-of-attack is modeled from boundary conditions, such that the inlet velocity is $U_\infty[\cos\alpha, 0, \sin\alpha]$.

Processing. The original authors publish 62 erroneous cases that were due to numerical instability, so we exclude these. We also negate OpenFOAM's shear stress convention. Given a bounding box $[(x_{min}, y_{min}, z_{min}), (x_{max}, y_{max}, z_{max})]$ of size $L = (L_x, L_y, L_z)$, the volume domain is cropped to $[x_{min} - 0.5L_x, x_{max} + 1.5L_x]$ (streamwise), $[y_{min} - 0.15L_y, y_{max} + 0.35L_y]$ (spanwise) and $[z_{min} - 0.35L_z, z_{max} + 0.35L_z]$ (vertical).

A.7 SUPERWING (YANG ET AL., 2026)

Overview. The dataset consists of 28,856 transonic solutions over 4239 kinked, double-swept wing planforms, each simulated at up to eight operating conditions, under 56 design variables per wing.

Data Generation. Samples are solved with ADflow as steady compressible RANS with a Spalart-Allmaras closure. The mesh is a structured grid that is extruded from the surface, with wall-normal layers at $y^+ \approx 1$, leading to the boundary layer being fully resolved. The solution uses a steady-state multigrid solver, with around 44k surface points and 3.1M volume cells per sample.

Parameters. Mach 0.75–0.90 at around fixed Re = 2.0e7 and T_∞ = 300 K, giving $U_\infty \approx$ 260–313 m/s, at angles of attack of $2°$ to $12°$.

Processing. The raw data frame has the chord along $+x$, span along z, and vertical along $+y$, with $+y$ being the suction side of the wing (low pressure); we therefore apply a signed permutation $(x, y, z) \mapsto (x, -z, y)$, which places negative C_p on the upper surface of a lifting wing. The volume mesh is highly dense due to being a wall-resolved simulation and around half of all volume points lie within 0.0085 chord of the wall and layer 0 reproduces the surface field exactly, however, we do not modify this. We prune wall-shear stress at 3σ per sample on all three components to remove outliers. Given a bounding box $[(x_{min}, y_{min}, z_{min}), (x_{max}, y_{max}, z_{max})]$ of size $L = (L_x, L_y, L_z)$, the volume domain is cropped to $[x_{min} - 0.25L_x, x_{max} + 0.75L_x]$ (streamwise), $[y_{min} - 0.20L_y, y_{max} + 0.02L_y]$ (spanwise, +y is base of wing) and $[z_{min} - 0.20L_z, z_{max} + 0.20L_z]$ (vertical).

A.8 DOUBLE-DELTA (SHEN & ALONSO, 2026)

Overview. The dataset consists of 2448 solutions over 272 parametric cranked-delta wings, each at nine angles of attack in 1° steps. It models leading-edge vortices and their breakdown at large angles of attack, and has large pressure/wall-shear deviations.

Data Generation. Samples are solved with SU2 as steady compressible RANS using SA-R, the Spalart-Allmaras model with a rotation and curvature correction. The convective scheme is JST with Green-Gauss reconstruction, integrated with implicit Euler at adaptive CFL; 57 runs required an AUSM scheme instead to converge and are flagged. Meshes are wall-resolved at a mean y^+ of 1.66, with around 1.18e5 surface elements and 6.96e6 volume cells of mixed tetrahedra, prisms and pyramids.

Parameters. Mach 0.3 with p_∞ = 71833.4 Pa, ρ_∞ = 0.878035 kg/m^3, $|U_\infty|$ = 101.53 m/s, q_∞ = 4525.54 Pa and T_∞ = 285 K, giving Re = 8.04e7 on a 16 m mean aerodynamic chord, at angles of attack of 11° to 19°.

Processing. There are a few duplicate runs that are excluded. Due to being a wall-resolved simulation, the majority ($> 90\%$) of mesh points lie in a thin slab around the wing, but we do not modify this. Surface pressure is pruned based on the stagnation bound $C_p \leq C_{p,0} = 1.0227$ at Mach 0.3 and a two-sided 3σ range, which removes around 2% of surface points. This clips the surface C_p minimum from -15.9 to about -5. The volume, which reaches $C_p = -9.7$, is not pruned. Splits are the dataset's native, geometry-disjoint ones: the 256 training-set geometries (2304 samples) against the 16 holdout geometries (144 samples). Given a bounding box $[(x_{min}, y_{min}, z_{min}), (x_{max}, y_{max}, z_{max})]$ of size $L = (L_x, L_y, L_z)$, the volume domain is cropped to $[x_{min} - 0.25L_x, x_{max} + 0.75L_x]$ (streamwise), $[y_{min} - 0.35L_y, y_{max} + 0.35L_y]$ (spanwise) and $[z_{min} - 0.2L_z, z_{max} + 0.2L_z]$ (vertical).

A.9 HILIFTAEROML (ASHTON ET AL., 2026)

Overview. The dataset consists of 1787 wall-modelled LES solutions over the NASA CRM-HL high-lift configuration. There are 180 geometry variants, described by eight design variables covering slat and flap deflections and gap multipliers inboard and outboard, each at ten angles of attack in 2° steps. It models high-lift aerodynamics through flow separation and is the largest dataset we use in terms of size and points per sample.

Data Generation. The geometry is a half-domain with a symmetry plane at $y = 0$ and positions in inches. Samples are solved with Fidelity Charles, an explicit unstructured finite-volume solver, second-order in space and third-order in time, using a dynamic Smagorinsky subgrid-scale model, whose coefficient is computed dynamically in space and time, and an equilibrium wall model solving simplified boundary-layer equations on an embedded near-wall grid. Voronoi meshes carry 300–500 M cells, reduced to 150–300 M on export. Statistics are gathered over 15 convective time units following a 10 unit transient below 12°, and over 30–200 units at the separated high-AoA conditions. Meshes are adapted per angle of attack to improve accuracy, so a geometry does not reuse one grid.

Parameters. Mach 0.2 at Re = 1.6e6, with a mean aerodynamic chord of 275.80 in (7.005 m) and a half-span of 1156.75 in.

SMART		SMART-IC		AB-UPT		Transolver++	
Parameter	Value	Parameter	Value	Parameter	Value	Parameter	Value
Model size	32.0M	Model size	32.4M	Model size	36.1M	Model size	29.1M
Hidden dim	384	Hidden dim	384	Hidden dim	384	Hidden dim	384
Heads	8	Heads	8	Heads	8	Heads	8
Enc./dec. blocks	12	Enc./dec. blocks	8	Shared blocks	ppscscs	Layers	12
Num latents	4096	Num latents	4096	Supernodes	1024	Slices	32
Geom. pts / block	4096	Geom. pts / block	4096	Geometry blocks	2	MLP ratio	4
–	–	Context pts / block	8192	Surf. / vol. blocks	3 / 3	–	–
–	–	–	–	Surf. / vol. anchors	4096 / 4096	–	–

Table 5: Model hyperparameters. All models use a hidden width of 384 and are matched at roughly 30M parameters. The optimizer is also constant across all models (Adam (β_1=0.9, β_2=0.999), learning rate 10^{-4}, StepLR γ=0.99 every 1k steps)

Processing. Given the large size of the dataset, we downsample the downloaded fields and meshes by a factor of two. Training and validation splits are grouped by geometry (90/10). Given a bounding box $[(x_{min}, y_{min}, z_{min}), (x_{max}, y_{max}, z_{max})]$ of size $L = (L_x, L_y, L_z)$, the volume domain is cropped to $[x_{min} - 0.15L_x, x_{max} + 0.5L_x]$ (+x is wake), $[y_{min} - 0.05L_y, y_{max} + 0.15L_y]$ (-y is symmetry plane) and $[z_{min} - L_z, z_{max} + L_z]$ (vertical)

A.10 SHIFT-CCA (Luminary Cloud, 2025a)

Overview. The dataset consists of 99 variants of a Group-5 UAV reference configuration representing collaborative combat aircraft concepts, morphed in root chord, panel break position, leading- and trailing-edge sweep and wing-tip geometry. All 99 geometries are distinct and share a single physical regime, and it models transonic flight with a shock on the wing.

Data Generation. Specific details are not disclosed, but we assume that samples are solved as steady RANS with a Spalart-Allmaras closure on the Luminary finite-volume solver. Meshes carry around 5.55 M volume points and 32.1 M tetrahedra, with about 1.5 M surface cells per case.

Parameters. Mach 0.72 at p_∞ = 18754 Pa, T_∞ = 216.65 K, ρ_∞ = 0.30155 kg/m^3 and U_∞ = 212.452 m/s, Re $\approx$ 3.1e7 and the angle of attack is 2°.

Processing. We prune surface shear at 4σ per sample on all three components, removing about 2% of points. Given a bounding box $[(x_{min}, y_{min}, z_{min}), (x_{max}, y_{max}, z_{max})]$ of size $L = (L_x, L_y, L_z)$, the volume domain is cropped to $[x_{min} - 0.25L_x, x_{max} + L_x]$ (+x is wake), $[y_{min} - 0.35L_y, y_{max} + 0.35L_y]$ and $[z_{min} - 0.2L_z, z_{max} + 0.2L_z]$ (vertical).

B Model Details

An overall figure comparing the main aspects of each architecture is shown in Figures 8 and 9. Additionally, we list the main hyperparameters of each model in Table 5. We briefly describe the main features of each architecture as well as any modifications we implement; however, for specific details we encourage readers to look at the original papers.

SMART. SMART (Hagnberger & Niepert, 2026) is a transformer-based surrogate model designed to approximate solutions on dense, unstructured meshes. Its central mechanism is to ensure that query point x attends to latent representations of the geometry G rather than to other queries, using cross attention. Avoiding interaction among queries has two benefits. First, it keeps computational cost under control: because queries attend only to the geometry encoding, whose discretization and sampling we choose, the cost no longer scales with the large number of query points. Second, it preserves discretization invariance, since the prediction at a given point does not change with the number of queries. We make a few slight improvements to the original SMART architecture. We

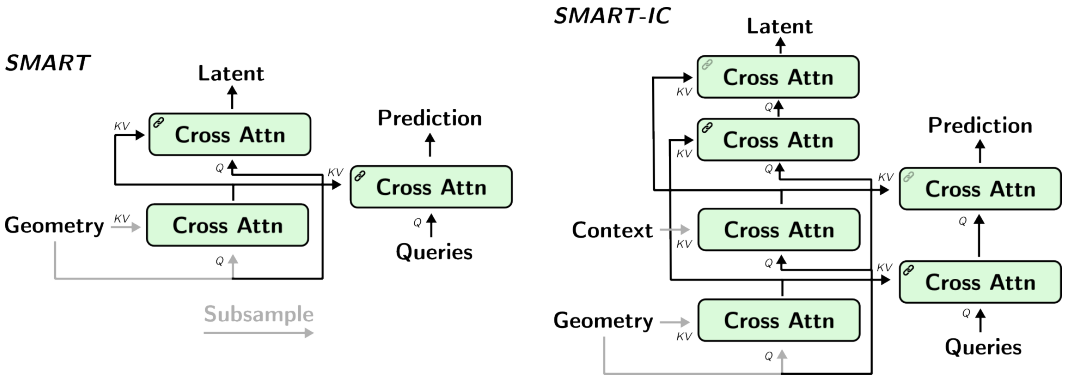

Figure 8: Schematic of the SMART model variants. Cross-attention is routed so that the encoder and decoder branches can share weights (link symbols). Gray arrows indicate subsampling. Simulation parameters θ enter through Feature-wise Linear Modulation (FiLM) layers (Perez et al., 2017)

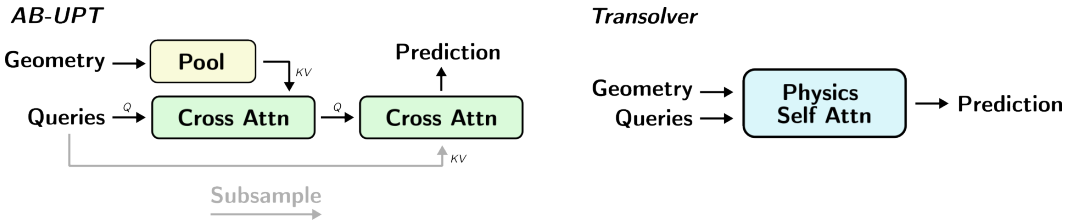

Figure 9: Schematics of AB-UPT and Transolver. Gray arrows indicate subsampling. For clarity, branches and weight sharing in AB-UPT, and slicing/unslicing in Transolver, are not shown. Simulation parameters θ enter through Adaptive LayerNorm (AdaLN) layers.

replace MLP layers with SwiGLU (Shazeer, 2020), as well as use separate output heads for the volume and surface fields.

SMART-IC. We build on the base SMART model to allow it to use an additional context simulation as input. We treat the context simulation as a surface and volume field, and draw samples from it to update the latent state in SMART. This latent state is additionally used in the decoder branch to propagate information to the queries. We assume that the surface field in the context contains information about the context geometry, so it is not additionally provided. However, both the context simulation parameters and the query simulation parameters θ_c, θ_q are provided.

AB-UPT. Anchored Branch Universal Physics Transformers (AB-UPT) (Alkin et al., 2025b) is another transformer-based surrogate model for approximating solutions on dense, unstructured meshes. Its main mechanism for reducing computational cost is to designate a subsampled set of query points as anchors; the remaining queries then obtain information from these anchors through cross-attention. Because the anchors are sampled randomly, each query depends only on a random subset of points rather than on all other queries. This makes predictions independent of the full query set and bounds the computational cost. In addition, global geometric information is incorporated through a pooling step followed by a cross-attention layer.

We make several modifications to improve model performance. First, we replace the radius pooling in the geometry branch with a pooling mechanism based on ball-tree decomposition (Zhdanov et al., 2025). Rather than selecting a random set of supernodes and pooling all points within a fixed radius of each, we partition the geometry point cloud into balls and pool the information within each ball to its centroid. Second, we replace the MLP layers with SwiGLU feed-forward layers.

Transolver++. Transolver++ (Luo et al., 2025) builds on Transolver (Wu et al., 2024), which introduced an efficient physics-attention mechanism for handling large input meshes. In this mechanism, a learned projection aggregates the input points onto a fixed set of slices, and self-attention is

Held-out Datasets:	AhmedML				SHIFT-Sub				SHIFT-CCA				HiLiftAeroML			
# FT Samples:	0	2	4	8	0	2	4	8	0	2	4	8	0	2	4	8
No Pretraining																
Random init.	0.68	0.35	0.33	0.25	0.65	0.32	0.28	0.30	0.69	0.28	0.27	0.26	0.82	0.63	0.63	0.61
Domain Experts																
DrivAerNet++	0.61	0.24	0.21	0.16	0.62	0.28	0.23	0.17	0.71	0.19	0.16	0.16	0.90	0.53	0.48	0.48
Emmi-Wing	0.74	0.24	0.22	0.19	0.68	0.29	0.24	0.22	0.70	0.20	0.18	0.17	1.02	0.60	0.54	0.53
SuperWing	1.61	0.23	0.22	0.18	1.35	0.30	0.24	0.22	0.69	0.18	0.16	0.15	2.76	0.57	0.54	0.53
DrivAerML	0.64	0.26	0.21	0.17	0.62	0.27	0.22	0.17	0.79	0.20	0.16	0.15	0.95	0.52	0.45	0.45
WindsorML	0.37	0.19	0.15	0.11	0.58	0.27	0.20	0.14	0.69	0.20	0.16	0.16	0.90	0.56	0.50	0.51
Double-Delta	0.73	0.24	0.24	0.19	0.68	0.27	0.23	0.19	0.76	0.14	0.12	0.12	0.84	0.54	0.44	0.42
Cross-Domain																
Joint Model	**0.31**	**0.14**	**0.11**	**0.10**	**0.53**	**0.23**	**0.18**	**0.13**	**0.61**	**0.13**	**0.11**	**0.11**	**0.82**	**0.48**	**0.42**	**0.39**

Table 6: Validation error for *AB-UPT* on four held-out target datasets. Rows indicate the model initialization; single-domain experts are identified by their pretraining dataset. Columns give the number of fine-tuning samples, with 0 denoting zero-shot evaluation. Green and red indicate the best and worst single-domain experts, respectively, and bold indicates the lowest overall error.

Held-out Datasets:	AhmedML				SHIFT-Sub				SHIFT-CCA				HiLiftAeroML			
# FT Samples:	0	2	4	8	0	2	4	8	0	2	4	8	0	2	4	8
No Pretraining																
Random Init.	0.68	0.36	0.34	0.30	0.64	0.31	0.29	0.27	0.68	0.22	0.21	0.20	0.81	0.61	0.59	0.58
Domain Experts																
DrivAerNet++	0.59	0.27	0.27	0.19	0.57	0.28	0.24	0.22	0.79	0.21	0.19	0.18	0.97	0.56	0.46	0.51
Emmi-Wing	0.70	0.27	0.24	0.21	0.66	0.29	0.24	0.22	0.65	0.21	0.18	0.18	1.01	0.58	0.50	0.49
SuperWing	0.73	0.36	0.31	0.23	0.71	0.29	0.25	0.25	0.78	0.19	0.16	0.16	0.96	**0.50**	0.43	0.41
DrivAerML	0.62	0.28	0.25	0.19	0.63	0.27	0.24	0.20	0.87	0.20	0.18	0.17	1.01	0.55	0.47	0.47
WindsorML	0.47	0.26	0.19	0.16	0.60	0.28	**0.23**	0.21	0.77	0.22	0.20	0.20	1.06	0.58	0.54	0.56
Double-Delta	0.70	0.37	0.34	0.24	0.66	0.29	0.25	0.23	0.76	0.19	0.15	0.15	**0.88**	0.58	0.48	0.47
Cross-Domain																
Joint Model	**0.40**	**0.15**	**0.12**	**0.11**	**0.55**	**0.26**	**0.23**	**0.18**	**0.62**	**0.17**	**0.14**	**0.13**	0.91	0.53	**0.42**	**0.39**

Table 7: Validation error for *Transolver++* on four held-out target datasets. Rows indicate the model initialization; single-domain experts are identified by their pretraining dataset. Columns give the number of fine-tuning samples, with 0 denoting zero-shot evaluation. Green and red indicate the best and worst single-domain experts, respectively, and bold indicates the lowest overall error.

then applied among these slices. While this reduces computational cost, the resulting representation remains tied to the discretization of the input.

In our implementation, we assume that the geometry information is contained in the surface queries, which are combined with the volume queries to form the model input. We additionally introduce adaptive layer normalization (AdaLN) to incorporate conditioning information into the model. Finally, as with the other models, we replace the MLP layers with SwiGLU feed-forward layers and use separate output heads for the surface and volume predictions.

C Additional Experiments

C.1 Different Architectures

Generalization to Unseen Datasets. We repeat the main experiments in Section 4.1 for AB-UPT and Transolver++ models. AB-UPT and Transolver++ models are instantiated to around the same size (∼30M parameters) and trained for the same budget (100k steps). After pretraining the expert and joint models, they are finetuned to the held-out datasets using $S \in \{0, 2, 4, 8\}$ finetuning samples. The resulting errors are shown in Table 6 for AB-UPT and Table 7 for Transolver++. Similar to the main text, we observe that cross-domain pretraining improves downstream performance com-

	Model	DrivAerNet	DrivAerML	Emmi-Wing	WindsorML	SuperWing	Double-Delta	Mean
Expert	SMART	**0.142**	**0.047**	0.030	**0.065**	**0.047**	**0.055**	**0.064**
	AB-UPT	0.148	0.057	0.030	0.068	0.058	0.064	0.071
	Transolver++	0.150	0.067	**0.029**	0.074	0.063	0.071	0.076
Joint	SMART	**0.173**	**0.134**	0.041	**0.074**	**0.082**	**0.090**	**0.099**
	AB-UPT	0.177	0.148	0.041	0.080	0.089	0.097	0.105
	Transolver++	0.184	0.166	0.044	0.097	0.107	0.124	0.120

Table 8: SMART, AB-UPT, and Transolver++ models, evaluated on the pretraining datasets. Note that each cell in the Expert block is a different model (for example, the DrivAerNet SMART expert), while each row in the Joint block is a single model evaluated across all datasets.

pared to any of the expert models or training from scratch. The overall trends and performance gains are also generally consistent with the main text.

Pretraining Performance. We additionally report the performance of all expert and joint models on the pretraining validation sets, in Table 8. In general, SMART seems to be the best model, but every model performs quite well and the differences between architectures are marginal. This also corroborates prior studies (Alkin et al., 2025a; Hagnberger & Niepert, 2026), where SMART/AB-UPT models are shown to outperform Transolver. However, the main benefit is that these cross attention-based architectures have query independence, such that they can be queried sequentially on a much denser mesh during inference without a large increase in error or memory.

C.2 Global Normalization

Overview. A frequently overlooked problem in finetuning is that accurate normalization statistics cannot be assumed in practice. Large existing datasets contain enough samples to compute reliable means and variances, but finetuning aims to use only a handful of samples, where these statistics might not be well-estimated. However, we can leverage physical priors to help address this. Most CFD datasets solve the Navier–Stokes equations, which admit well-known non-dimensional quantities. These quantities are anchored to reference quantities set by the simulation, most commonly the freestream velocity, pressure, and density. Because these reference values are known at deployment, we use them to compute non-dimensional coefficients and variables, such as the coefficient of pressure C_p, skin friction coefficient C_f, and non-dimensional velocity u^*:

$$C_p = \frac{p - p_\infty}{\frac{1}{2}\rho_\infty U_\infty^2}, \quad C_f = \frac{\vec{\tau}_w}{\frac{1}{2}\rho_\infty U_\infty^2}, \quad u^* = \frac{\vec{u}}{U_\infty}$$

where ρ_∞ is the freestream density, p_∞ is the freestream pressure, and U_∞ is the freestream velocity magnitude. These quantities greatly help to align different ranges across datasets. For example, AhmedML has a maximum velocity magnitude $|\mathbf{u}|$ of around 1m/s, while Emmi Wing has a maximum velocity magnitude $|\mathbf{u}|$ of around 300 m/s. After non-dimensionalization, the velocity magnitude across all datasets collapses to an approximate range $|\mathbf{u}^*| \in [0, 1.5]$, which can be seen in Figure 7. Similarly, the raw volumetric pressure can vary widely across datasets, but after non-dimensionalization, the coefficient of pressure collapses to an approximate range $C_p \in [-1, 1]$ (can be much more negative). These ranges are sometimes even enforced by conservation laws; for example, the coefficient of pressure is bounded in incompressible flows ($C_p < 1$); values above this limit would require negative velocity magnitudes $|\mathbf{u}| < 0$.

We therefore perform an additional set of studies where we train models using global normalization statistics, in which each variable is scaled by the same constant across all datasets. Different constants for each variable are still used to bring different quantities into a comparable range; the skin friction coefficient C_f, for example, is typically orders of magnitude smaller than the pressure coefficient or velocity. After non-dimensionalization, the velocity is additionally expressed as a deficit relative to the freestream ($u' = (\vec{u} - \vec{U}_\infty)/U_\infty$) and then each output field is divided by its scaling factor: $f_{C_p} = 0.33$, $f_{C_f} = 0.002$, and $f_{u'} = 0.2$. For vector quantities, the same factor is applied to every component. These factors were computed on the pretraining set, excluding the finetuning data, so that the resulting non-dimensional fields have approximately unit scale.

Quantity	Dataset-specific	Global
Pressure coefficient	$(C_p - \mu_d^{(C_p)})/\sigma_d^{(C_p)}$	C_p/f_{C_p}, $f_{C_p} = 0.33$
Skin friction coefficient	$(C_f - \mu_d^{(C_f)})/\sigma_d^{(C_f)}$	C_f/f_{C_f}, $f_{C_f} = 0.002$
Velocity	$(u^* - \mu_d^{(u^*)})/\sigma_d^{(u^*)}$, $u^* = \vec{u}/U_\infty$	$u'/f_{u'}$, $u' = (\vec{u} - \vec{U}_\infty)/U_\infty$, $f_{u'} = 0.2$
Simulation parameters	$(\theta - \theta_{\min})/(\theta_{\max} - \theta_{\min})$	$(\theta - \theta_{\min})/(\theta_{\max} - \theta_{\min})$
Geometry and queries	$(G - \bar{c}_G)/s_d$, $(x - \bar{c}_G)/s_d$	$(G - \bar{c}_G)/s_d$, $(x - \bar{c}_G)/s_d$
Estimated from simulation	μ_d, σ_d for every field	None

Table 9: Comparison of dataset-specific and global normalization. The subscript d denotes a quantity computed separately for each dataset.

Held-out Datasets:	AhmedML				SHIFT-Sub				SHIFT-CCA				HiLiftAeroML			
# FT Samples:	0	2	4	8	0	2	4	8	0	2	4	8	0	2	4	8
No Pretraining																
Random Init.	0.82	0.42	0.40	0.33	0.82	0.38	0.31	0.30	0.85	0.22	0.21	0.21	0.90	0.65	0.65	0.64
Domain Experts																
DrivAerNet++	0.53	0.15	0.13	0.12	0.89	0.19	0.12	0.09	0.89	0.13	0.11	0.10	0.94	0.55	0.47	0.46
Emmi-Wing	0.71	0.20	0.20	0.19	0.67	0.24	0.17	0.15	0.68	0.14	0.12	0.12	0.82	0.54	0.46	0.46
SuperWing	0.76	0.23	0.24	0.19	0.72	0.22	0.16	0.13	0.65	0.12	0.11	0.11	0.83	**0.46**	0.37	0.38
DrivAerML	0.51	0.18	0.16	0.14	0.83	0.17	**0.10**	**0.08**	0.88	0.13	**0.10**	0.10	0.94	0.51	0.44	0.45
WindsorML	0.57	0.18	0.13	0.11	1.43	0.23	0.14	0.12	1.13	0.15	0.13	0.13	0.88	0.57	0.52	0.51
Double-Delta	0.82	0.20	0.20	0.17	0.83	0.22	0.18	0.13	0.68	0.11	0.11	0.10	0.86	0.49	0.37	0.36
Cross-Domain																
Joint Model	**0.50**	**0.13**	**0.11**	**0.09**	**0.64**	**0.14**	**0.10**	0.09	**0.55**	**0.10**	**0.10**	**0.09**	**0.75**	**0.46**	**0.36**	**0.34**

Table 10: Validation error for *SMART (with global normalization)* on four held-out target datasets. Rows indicate the model initialization; single-domain experts are identified by their pretraining dataset. Columns give the number of fine-tuning samples, with 0 denoting zero-shot evaluation. Green and red indicate the best and worst single-domain experts, respectively, and bold indicates the lowest overall error.

Normalization for simulation parameters θ, geometries G, and queries x are left unchanged. Simulation parameters are already min–max scaled using fixed bounds shared across all datasets, $\theta_{\min} = [0, -20^\circ]$ and $\theta_{\max} = [1, 20^\circ]$, corresponding to $\mathrm{Ma} \in [0, 1]$ and $\alpha \in [-20^\circ, 20^\circ]$. This was found to be important such that the same normalized value corresponds to the same operating condition across all datasets. We choose dataset-specific statistics for the geometry because the centroid $\bar{c}_G$ and characteristic bounding box or length scale are design parameters and therefore are known at deployment. Each input geometry is centered at the origin and scaled by a dataset-specific factor s_d. Table 9 provides a summary for how normalization is typically done for output fields (means/variances computed over a dataset) and a global normalization.

Results. We repeat the main experiments of Section 4.1 for expert and joint models trained with global normalization. Zero- and few-shot generalization results are reported in Table 10, and pretraining results are in Table 11. The overall trend still holds, with the joint model matching or outperforming all baselines in 15 of the 16 settings. Few-shot errors of the pretrained models change little compared to dataset-specific normalization (Table 2), and although the joint model's absolute zero-shot error rises on three of the four held-out datasets, it now achieves the lowest zero-shot error on all four.

During pretraining the two normalization schemes are also very similar. For expert models, the choice of normalization has no measurable effect, which is expected. Each expert sees only a single dataset, so any constant shift or scale is readily absorbed during training. For the joint model, the effect is likewise negligible during pretraining. Global normalization may even be advantageous for the output fields, since it preserves the physical meaning of each variable across datasets. For example, $C_p = 1$ consistently corresponds to a stagnation point, and $C_f = 0$ (vanishing wall-shear stress) consistently indicates flow separation, regardless of the dataset. Overall, these results show

Model	Normalization	DrivAerNet	DrivAerML	Emmi-Wing	WindsorML	SuperWing	Double-Delta	Mean
Expert	Dataset-specific	0.142	0.047	0.030	0.067	0.048	0.057	0.065
	Global	0.142	0.046	0.030	0.067	0.047	0.058	0.065
Joint	Dataset-specific	0.173	0.134	0.043	0.074	0.082	0.090	0.099
	Global	0.172	0.132	0.041	0.074	0.079	0.091	0.098

Table 11: Comparison of dataset-specific and global normalization during pretraining. Losses are computed in the de-normalized space (raw C_p, C_f, u^*). Note that each cell in the Expert block is a different model, while each row in the Joint block is a single model evaluated across all datasets.

Held-out Datasets:	AhmedML				SHIFT-Sub				SHIFT-CCA				HiLiftAeroML			
# FT Samples:	0	2	4	8	0	2	4	8	0	2	4	8	0	2	4	8
No Pretraining																
Random Init.	0.77	0.57	0.54	0.40	0.83	0.53	0.49	0.47	0.87	0.53	0.45	0.46	0.87	0.76	0.75	0.76
Domain Experts																
DrivAerNet++	0.57	0.17	0.14	0.12	**0.64**	0.31	0.23	0.19	0.88	0.34	0.26	0.24	0.97	0.65	0.57	0.49
Emmi-Wing	0.81	0.22	0.21	0.19	0.77	0.38	0.31	0.27	0.77	0.28	0.26	0.26	0.98	0.62	0.54	0.57
SuperWing	0.86	0.24	0.22	0.19	0.80	0.38	0.34	0.27	0.93	0.26	0.25	0.24	1.11	0.60	0.49	0.47
DrivAerML	0.55	0.17	0.18	0.14	0.70	0.29	0.22	**0.18**	0.84	0.32	0.24	0.23	0.93	0.70	0.56	0.46
WindsorML	0.45	0.20	0.14	0.12	0.69	0.35	0.29	0.24	0.87	0.37	0.29	0.28	0.92	0.71	0.60	0.54
Double-Delta	0.79	0.19	0.17	0.17	0.82	0.36	0.31	0.23	0.85	0.24	0.22	0.22	**0.89**	0.55	**0.40**	0.37
Cross-Domain																
Joint	**0.34**	**0.12**	**0.10**	**0.09**	0.66	**0.25**	**0.21**	**0.18**	**0.70**	**0.22**	**0.20**	**0.20**	**0.89**	**0.52**	**0.40**	**0.36**

Table 12: Validation error for *SMART (evaluated on simulation mesh)* on four held-out target datasets. Rows indicate the model initialization; single-domain experts are identified by their pretraining dataset. Columns give the number of fine-tuning samples, with 0 denoting zero-shot evaluation. Green and red indicate the best and worst single-domain experts, respectively, and bold indicates the lowest overall error.

that global normalization can perform well, but the choice of normalization is still important and deserves more study. Given the large number of mesh points per simulation, it may still be possible to get sufficient estimates of means/variances from a few finetuning samples.

C.3 Discretization

Overview. In our main experiments, we sample query points approximately uniformly in space rather than drawing them at random from the simulation mesh. This is a design choice we made to keep the discretization more consistent across domains. Furthermore, uniform spatial sampling also better reflects deployment conditions: meshing is expensive, predominantly CPU-based, and still relies on human expertise to refine, so a simulation mesh is generally unavailable at inference time. However, this is generally not an issue, as prior studies have already shown that sampling the queries differently during inference does not significantly affect errors due to the discretization invariance of AB-UPT and SMART. For example, training on the simulation mesh, then querying the model on the CAD mesh has already been shown to work well (Alkin et al., 2025a;b; Hagnberger & Niepert, 2026). Nevertheless, we also perform a similar study by querying models trained on the more spatially uniform mesh on the simulation mesh. In some cases, querying on the simulation mesh may better resolve steep gradients, for example in the boundary layer or in regions of adaptive mesh refinement. However, as noted above, this mesh is usually not available before the simulation has been run.

Results. We replicate the main results in Section 4.1, but instead query the finetuned models on the simulation mesh. Zero- and few-shot errors for the experts and joint models are given in Table 12. We can see that the error for all models tends to rise; this is because spatially uniform samples tend to resolve more of the freestream, which is easier to model. However, we can see that the overall trend is still consistent, where the joint model has better zero- and few-shot performance,

Model	DrivAerNet	Emmi-Wing	WindsorML	Double-Delta	DrivAerML	SuperWing
Joint (Base)	0.174	0.043	0.075	0.091	0.134	0.082
Joint (121M)	**0.150**	0.039	0.072	0.070	0.074	0.074
Joint (6xSteps)	**0.150**	**0.029**	**0.069**	**0.063**	**0.073**	**0.041**

Table 13: Validation error of the joint model across the pretraining datasets, either with an increased model size (121M) or increased gradient budget (6xSteps).

Held-out Datasets:	AhmedML				SHIFT-Sub				SHIFT-CCA				HiLiftAeroML			
# FT Samples:	0	2	4	8	0	2	4	8	0	2	4	8	0	2	4	8
Joint (Base)	**0.31**	**0.12**	0.10	0.09	**0.52**	**0.14**	**0.10**	0.09	**0.52**	0.11	0.10	0.10	**0.90**	0.46	0.34	0.33
Joint (121M)	**0.31**	**0.12**	**0.09**	**0.08**	0.53	0.16	0.11	**0.08**	0.55	**0.09**	**0.08**	**0.08**	0.92	0.45	**0.32**	**0.28**
Joint (6xSteps)	0.32	**0.12**	0.10	**0.08**	0.61	0.16	**0.10**	0.09	0.55	0.10	0.09	0.09	1.11	**0.44**	**0.32**	0.30

Table 14: Validation error on four held-out target datasets. Each row is a different joint model, either with increased model size (121M) or increased gradient budget (6xSteps). Columns give the number of fine-tuning samples, with 0 denoting zero-shot evaluation.

which validates that cross-domain pretraining is still helpful, regardless of the discretization. It also serves to corroborate prior studies that show discretization invariance for SMART/AB-UPT.

C.4 Scaling Model Size or Compute

We present additional results on the effects of scaling model size and pretraining compute (number of gradient steps), both on pretraining (Table 13) and on generalization to unseen datasets (Table 14). As expected, the pretraining loss decreases consistently with both larger models and longer training. However, these gains do not generally carry over to unseen datasets: beyond a certain point, further increasing either model size or the number of gradient steps can raise the zero-shot error, suggesting that the model begins to overfit to the pretraining distribution. Few-shot performance likewise does not consistently improve with scale, although larger models tend to be slightly more sample-efficient. These results suggest that generating more CFD data is the most promising way to improve generalization, followed by increasing model size and then training compute.

C.5 In-Context Pretraining

In time-dependent PDE settings, the predominant approach for cross-domain models is to infer the underlying dynamics from a set of paired context frames. To test whether a similar mechanism is needed for steady-state problems, we train a joint in-context model across domains (SMART-IC). Alongside the standard inputs (the query geometry G_q, query points x_q, and query parameters θ_q), SMART-IC receives a solved context simulation drawn from the same dataset as the query. This context consists of the field values $f_c(x_c)$ at a set of context points x_c and the corresponding simulation parameters θ_c, so the model's prediction is $f_\phi(x_q; G_q, \theta_q, f_c(x_c), \theta_c)$. We do not pass the context geometry G_c explicitly, since it is implicitly encoded by the surface points on which the context field $f_c(x_c)$ is defined.

Table 15 compares the pretraining performance of SMART-IC with that of the base model, SMART. Providing a context simulation yields no benefit, and in most cases the error is slightly higher. This is expected for steady-state problems: the forward problem is well-posed given only the query geometry G_q and parameters θ_q, so the output $f_q(x_q)$ is fully determined by these inputs. Additional context would help only if unobserved factors, such as differences in closure model or solver, varied across similar model inputs and left the problem ill-defined. This does not seem to occur in current, steady-state CFD datasets. The extra input therefore carries no useful information and, at worst, adds variance that increases the error. Consistent with this, we found that SMART-IC ignores the context entirely: zeroing out the context at inference time leaves its predictions unchanged.

As a whole, these results suggest that cross-domain CFD surrogates may require different strategies from those developed for time-dependent PDEs. In temporal problems, context is useful because

Model	DrivAerNet	Emmi-Wing	WindsorML	Double-Delta	DrivAerML	SuperWing
Joint (Base)	0.174	0.043	0.075	0.091	0.134	0.082
Joint (IC)	0.176	0.075	0.076	0.098	0.144	0.098

Table 15: Pretraining validation error of the joint base model (SMART) and the in-context model (SMART-IC) on each dataset. SMART-IC additionally receives a solved context simulation from the same dataset as the query. Providing context does not reduce the error on any dataset.

Held-out Datasets:	AhmedML				SHIFT-Sub				SHIFT-CCA				HiLiftAeroML			
# FT Samples:	0	2	4	8	0	2	4	8	0	2	4	8	0	2	4	8
No Pretraining																
Random init.	0.82	0.74	0.65	0.25	0.88	0.53	0.39	0.37	0.97	0.18	0.16	0.16	0.96	0.68	0.69	0.68
Domain Experts																
DrivAerNet++	0.65	0.21	0.17	0.14	0.87	0.36	0.20	0.13	0.96	0.13	0.08	0.07	1.08	0.59	0.53	0.52
Emmi-Wing	0.85	0.24	0.22	0.20	0.90	0.40	0.30	0.22	0.89	0.12	0.10	0.10	1.12	0.58	0.51	0.52
SuperWing	0.82	0.28	0.25	0.20	0.89	0.37	0.24	0.19	0.97	0.11	0.10	0.09	1.12	0.55	0.48	0.47
DrivAerML	0.61	0.21	0.19	0.17	0.87	0.33	0.18	0.12	0.92	0.11	**0.07**	**0.06**	1.02	0.61	0.52	0.52
WindsorML	0.49	0.21	0.15	0.12	**0.73**	0.32	0.22	0.15	1.02	0.15	0.09	0.09	1.04	0.62	0.57	0.56
Double-Delta	0.86	0.22	0.21	0.18	0.92	0.39	0.32	0.18	1.02	**0.08**	**0.07**	0.07	**0.87**	0.52	0.40	0.37
Cross-Domain																
Joint Model	**0.37**	**0.14**	**0.11**	**0.09**	0.78	**0.22**	**0.14**	**0.11**	**0.76**	**0.08**	**0.07**	0.07	**0.87**	**0.50**	**0.37**	**0.33**

Table 16: Validation error in *surface pressure coefficient* C_p on four held-out target datasets. Rows indicate the model initialization; single-domain experts are identified by their pretraining dataset. Columns give the number of fine-tuning samples, with 0 denoting zero-shot evaluation. Green and red indicate the best and worst single-domain experts, respectively, and bold indicates the lowest overall error.

domains differ mainly in their dynamics (for example, in the PDE terms or coefficients), which the model can infer from observed context frames. In steady-state CFD, by contrast, every domain is governed by the same Navier–Stokes equations, and the variation across domains lies instead in the geometry and boundary conditions. Future cross-domain methods may therefore benefit more from better representations of geometric and boundary-condition variability than from in-context mechanisms.

C.6 Variable-Specific Errors

For the main finetuning experiment in Section 4.1, we also report per-variable errors in place of the averaged error. Tables 16, 17, 18, and 19 give the NRMSE for surface C_p, surface C_f, volume C_p, and volume $\mathbf{u}^*$, respectively. Surface fields are generally harder to predict than volume fields, since much of the volume lies in the freestream, which is generally smooth. Nonetheless, the main trends hold across all variables: the joint model generally outperforms both the domain experts and the models trained from scratch.

D Additional Visualizations

For the main set of experiments in Section 4.1, we plot model predictions after zero-shot and few-shot evaluation ($S = 2$). These are 16 figures in total; for each finetuning dataset the surface C_p, surface C_{f_x}, volume C_p, and volume magnitude $|\mathbf{u}^*|$ is plotted.

Held-out Datasets:	AhmedML				SHIFT-Sub				SHIFT-CCA				HiLiftAeroML			
# FT Samples:	0	2	4	8	0	2	4	8	0	2	4	8	0	2	4	8
No Pretraining																
Random init.	0.65	0.61	0.60	0.49	0.41	0.35	0.34	0.32	0.41	0.30	0.31	0.31	0.77	0.73	0.74	0.73
Domain Experts																
DrivAerNet++	0.64	0.17	0.14	0.12	0.45	0.19	0.13	0.12	0.44	0.20	0.18	0.18	0.86	0.62	0.56	0.55
Emmi-Wing	0.75	0.19	0.19	0.18	0.49	0.19	0.16	0.15	0.39	0.19	0.18	0.18	1.39	0.63	0.59	0.61
SuperWing	0.72	0.23	0.21	0.18	0.46	0.19	0.15	0.15	0.39	0.19	0.18	0.18	1.13	0.60	0.58	0.58
DrivAerML	0.56	0.17	0.15	0.14	0.42	0.17	**0.12**	**0.11**	0.46	0.19	**0.17**	**0.17**	0.86	0.64	0.55	0.55
WindsorML	0.41	0.17	0.13	0.11	**0.37**	0.17	0.14	0.12	0.39	0.21	0.19	0.19	**0.81**	0.66	0.60	0.60
Double-Delta	0.68	0.18	0.17	0.16	0.41	0.19	0.17	0.14	0.45	0.18	0.18	0.18	0.82	0.58	0.47	0.46
Cross-Domain																
Joint Model	**0.34**	**0.11**	**0.10**	**0.08**	0.38	**0.14**	**0.12**	**0.11**	**0.30**	**0.17**	**0.17**	**0.17**	0.84	**0.56**	**0.44**	**0.42**

Table 17: Validation error in *surface skin-friction coefficient* C_f on four held-out target datasets. Rows indicate the model initialization; single-domain experts are identified by their pretraining dataset. Columns give the number of fine-tuning samples, with 0 denoting zero-shot evaluation. Green and red indicate the best and worst single-domain experts, respectively, and bold indicates the lowest overall error.

Held-out Datasets:	AhmedML				SHIFT-Sub				SHIFT-CCA				HiLiftAeroML			
# FT Samples:	0	2	4	8	0	2	4	8	0	2	4	8	0	2	4	8
No Pretraining																
Random init.	0.98	0.57	0.47	0.36	1.00	0.42	0.33	0.36	0.99	0.21	0.21	0.21	0.98	0.62	0.63	0.63
Domain Experts																
DrivAerNet++	0.63	0.24	0.21	0.18	0.85	0.24	0.13	0.09	1.14	0.14	0.09	0.08	1.48	0.53	0.46	0.46
Emmi-Wing	1.01	0.28	0.30	0.27	1.01	0.29	0.21	0.18	0.91	0.13	0.10	0.10	1.13	0.51	0.46	0.47
SuperWing	1.04	0.30	0.29	0.25	1.08	0.26	0.17	0.15	0.96	0.11	0.10	0.10	1.05	0.49	0.43	0.41
DrivAerML	0.67	0.25	0.27	0.23	0.85	0.21	0.12	0.08	1.03	0.14	0.08	**0.07**	1.13	0.53	0.45	0.45
WindsorML	0.45	0.21	0.17	0.14	0.93	0.22	0.14	0.10	1.19	0.17	0.12	0.11	1.50	0.54	0.52	0.49
Double-Delta	1.00	0.27	0.27	0.23	1.01	0.29	0.23	0.14	1.00	0.09	0.08	0.08	**1.02**	0.46	0.35	0.33
Cross-Domain																
Joint Model	**0.36**	**0.18**	**0.14**	**0.13**	**0.73**	**0.15**	**0.09**	**0.07**	**0.84**	**0.08**	**0.07**	**0.07**	1.09	**0.43**	**0.32**	**0.29**

Table 18: Validation error in *volume pressure coefficient* C_p on four held-out target datasets. Rows indicate the model initialization; single-domain experts are identified by their pretraining dataset. Columns give the number of fine-tuning samples, with 0 denoting zero-shot evaluation. Green and red indicate the best and worst single-domain experts, respectively, and bold indicates the lowest overall error.

Held-out Datasets:	AhmedML				SHIFT-Sub				SHIFT-CCA				HiLiftAeroML			
# FT Samples:	0	2	4	8	0	2	4	8	0	2	4	8	0	2	4	8
No Pretraining																
Random init.	0.26	0.26	0.25	0.25	0.26	0.25	0.25	0.25	0.35	0.24	0.25	0.25	0.53	0.49	0.50	0.49
Domain Experts																
DrivAerNet++	0.19	0.09	0.07	0.06	0.27	0.08	**0.07**	**0.06**	0.39	0.12	0.11	0.11	0.97	0.38	0.33	0.33
Emmi-Wing	0.29	0.12	0.13	0.12	0.24	0.09	0.09	0.08	0.25	0.12	0.11	0.12	0.77	0.37	0.35	0.36
SuperWing	0.70	0.14	0.13	0.13	0.69	0.09	0.09	0.09	0.65	0.11	0.11	0.11	1.74	0.37	0.34	0.33
DrivAerML	0.21	0.09	0.10	0.09	0.29	0.08	**0.07**	**0.06**	0.44	0.12	0.11	0.11	0.95	0.40	0.33	0.32
WindsorML	0.18	0.08	0.06	0.06	0.32	0.13	0.11	0.09	0.41	0.16	0.15	0.15	0.80	0.44	0.35	0.35
Double-Delta	0.29	0.11	0.11	0.10	0.26	0.08	0.08	0.07	0.30	**0.09**	**0.09**	**0.09**	**0.73**	0.37	0.28	0.27
Cross-Domain																
Joint Model	**0.15**	**0.07**	**0.05**	**0.05**	**0.18**	**0.07**	**0.07**	**0.06**	**0.20**	0.10	**0.09**	0.10	0.79	**0.36**	**0.25**	**0.25**

Table 19: Validation error in *volume velocity* $\mathbf{u}^*$ on four held-out target datasets. Rows indicate the model initialization; single-domain experts are identified by their pretraining dataset. Columns give the number of fine-tuning samples, with 0 denoting zero-shot evaluation. Green and red indicate the best and worst single-domain experts, respectively, and bold indicates the lowest overall error.

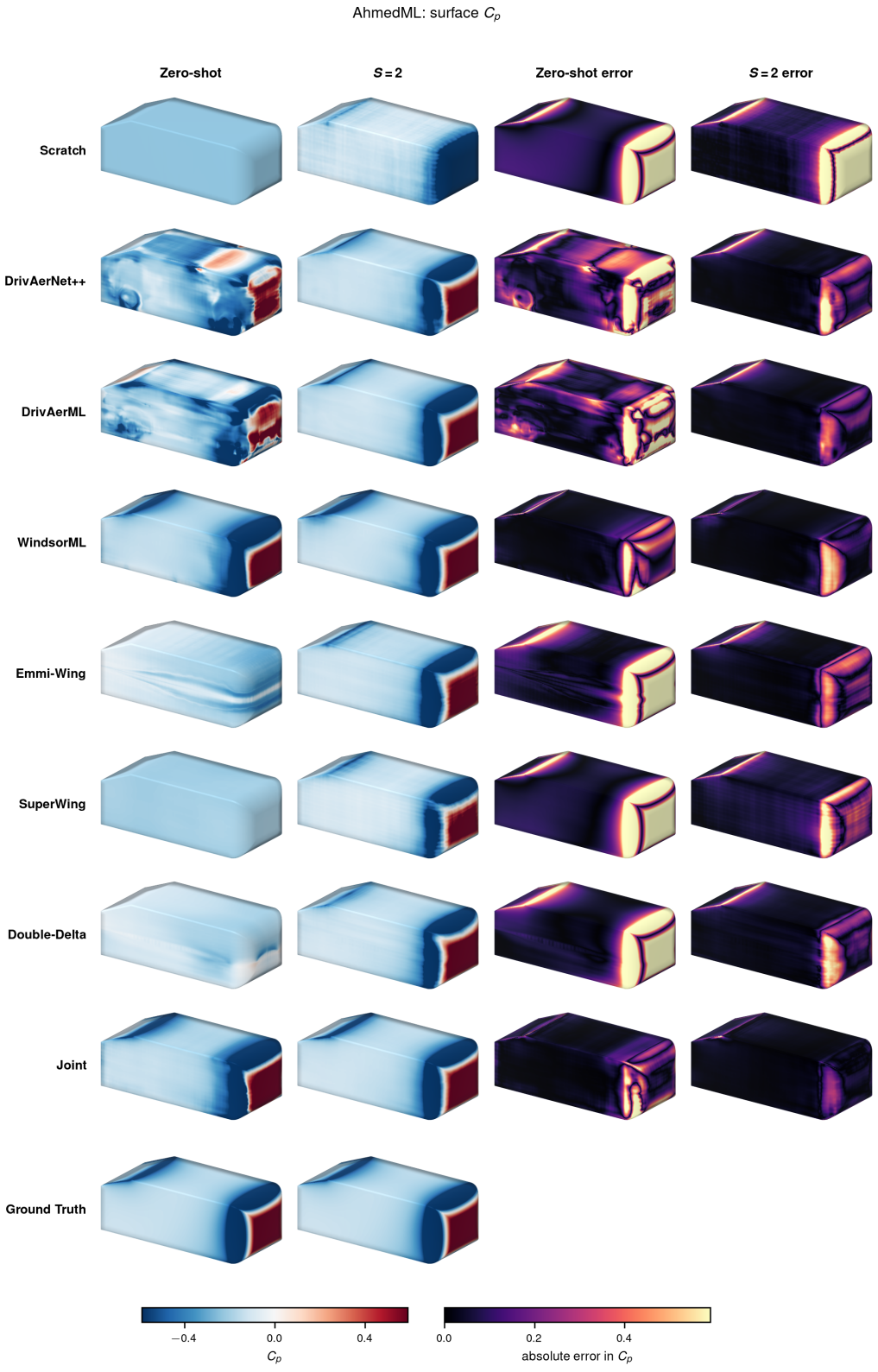

Figure 10: Model predictions of *surface* C_p on AhmedML.

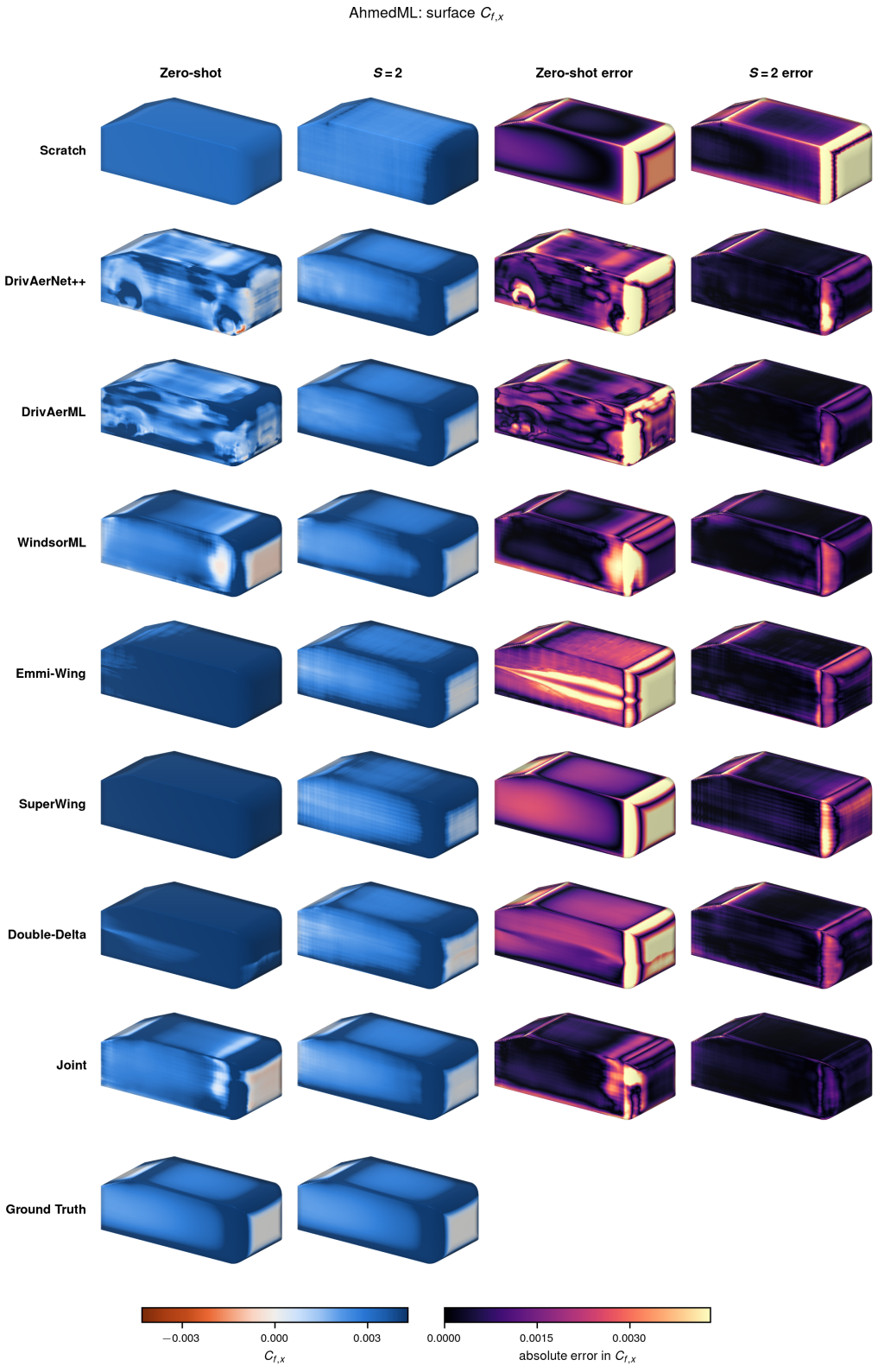

Figure 11: Model predictions of *surface* C_{f_x} on AhmedML.

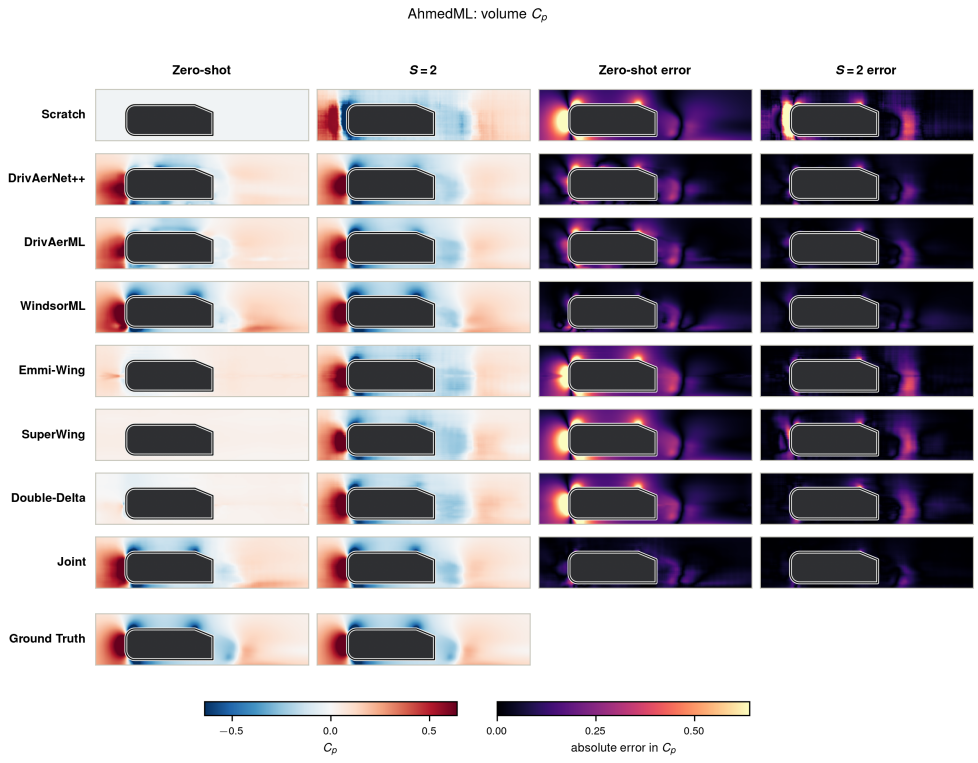

Figure 12: Model predictions of *volume* C_p on AhmedML.

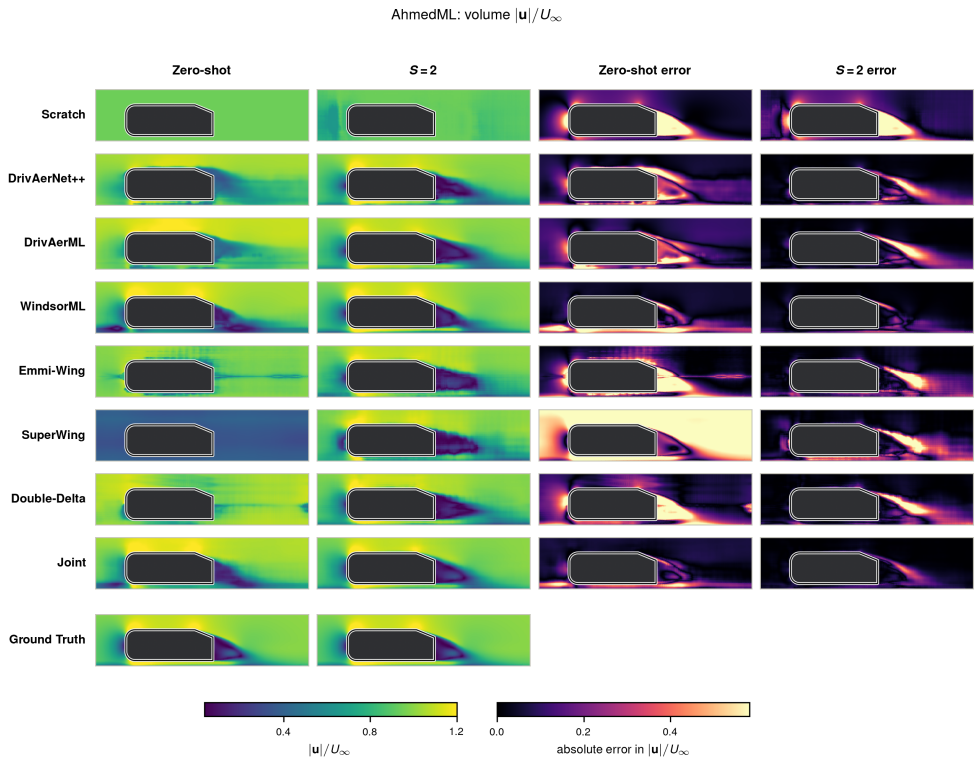

Figure 13: Model predictions of *volume* $|\mathbf{u}^*|$ on AhmedML.

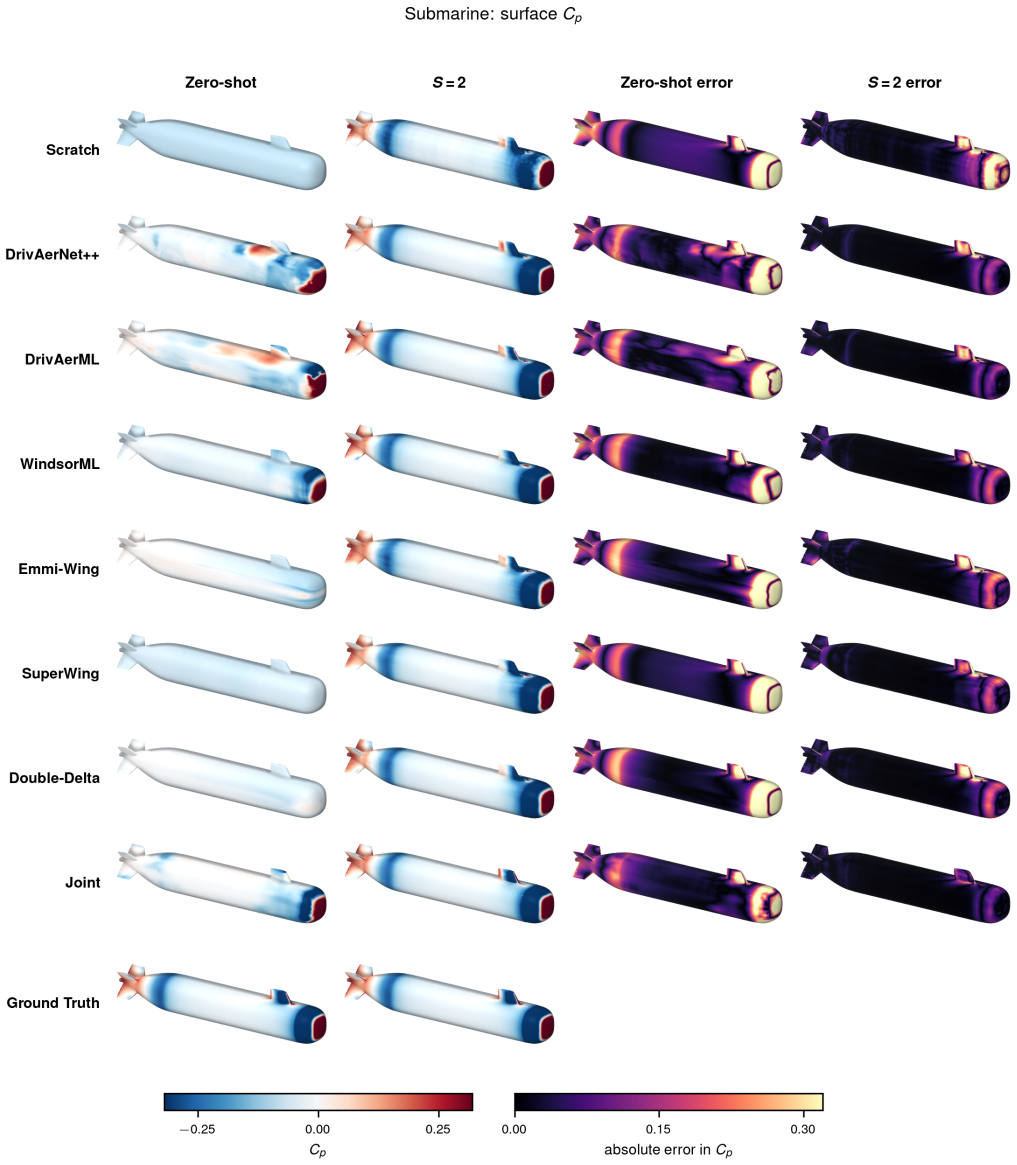

Figure 14: Model predictions of *surface* C_p on SHIFT-Submarine.

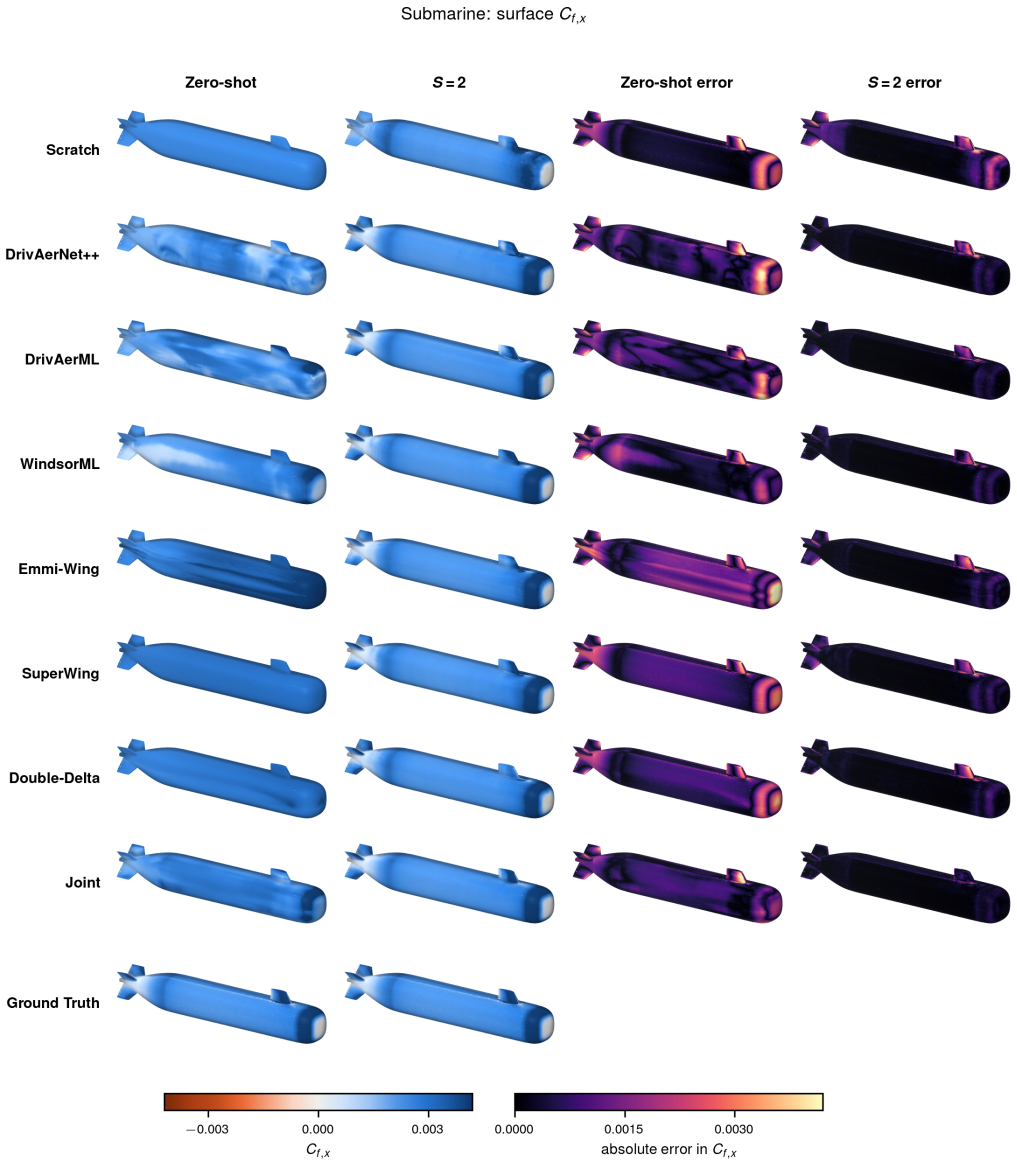

Figure 15: Model predictions of *surface* C_{f_x} on SHIFT-Submarine.

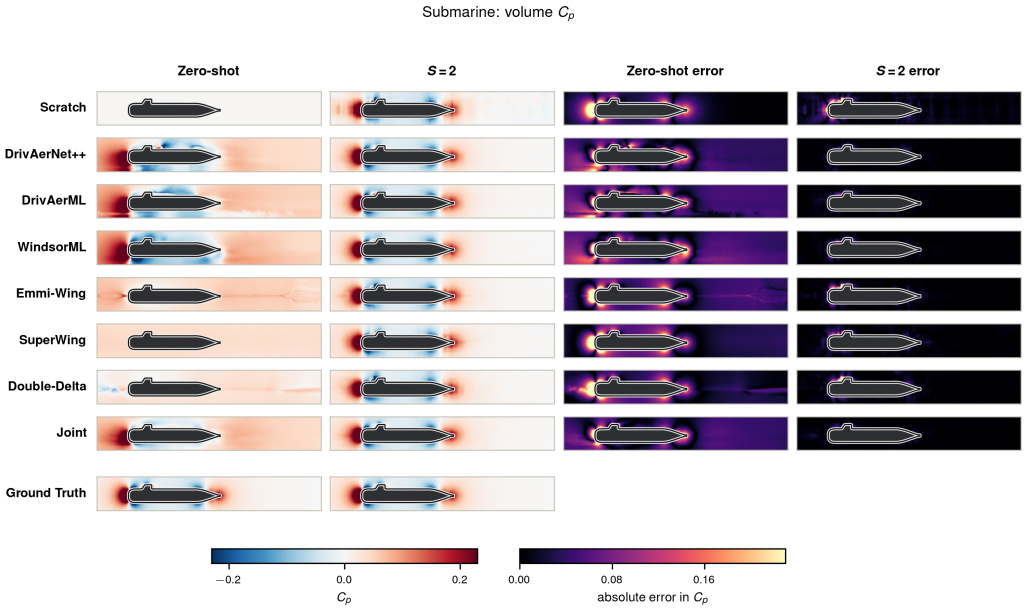

Figure 16: Model predictions of *volume* C_p on SHIFT-Submarine.

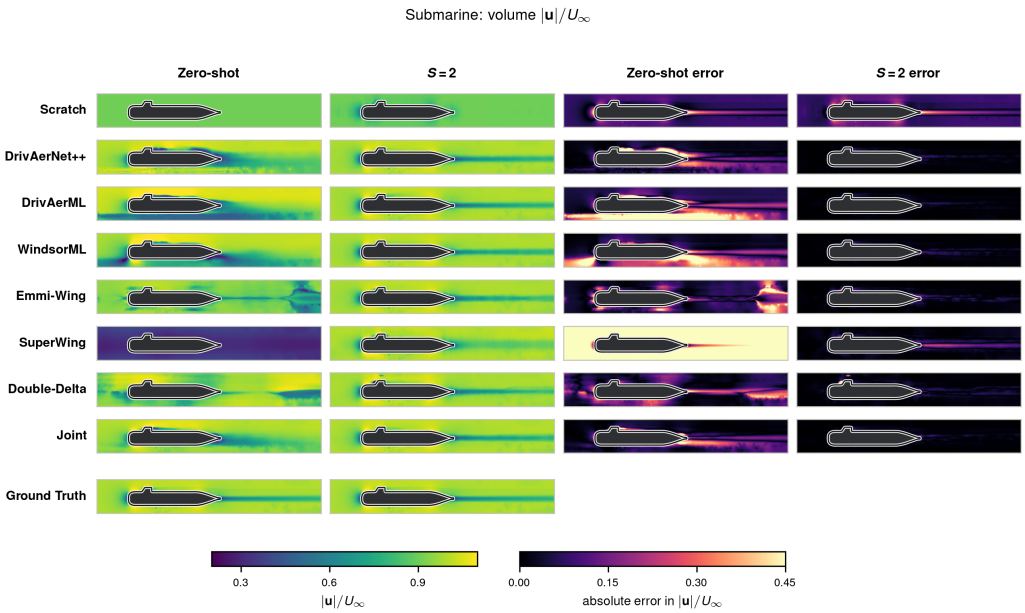

Figure 17: Model predictions of *volume* $|\mathbf{u}^*|$ on SHIFT-Submarine.

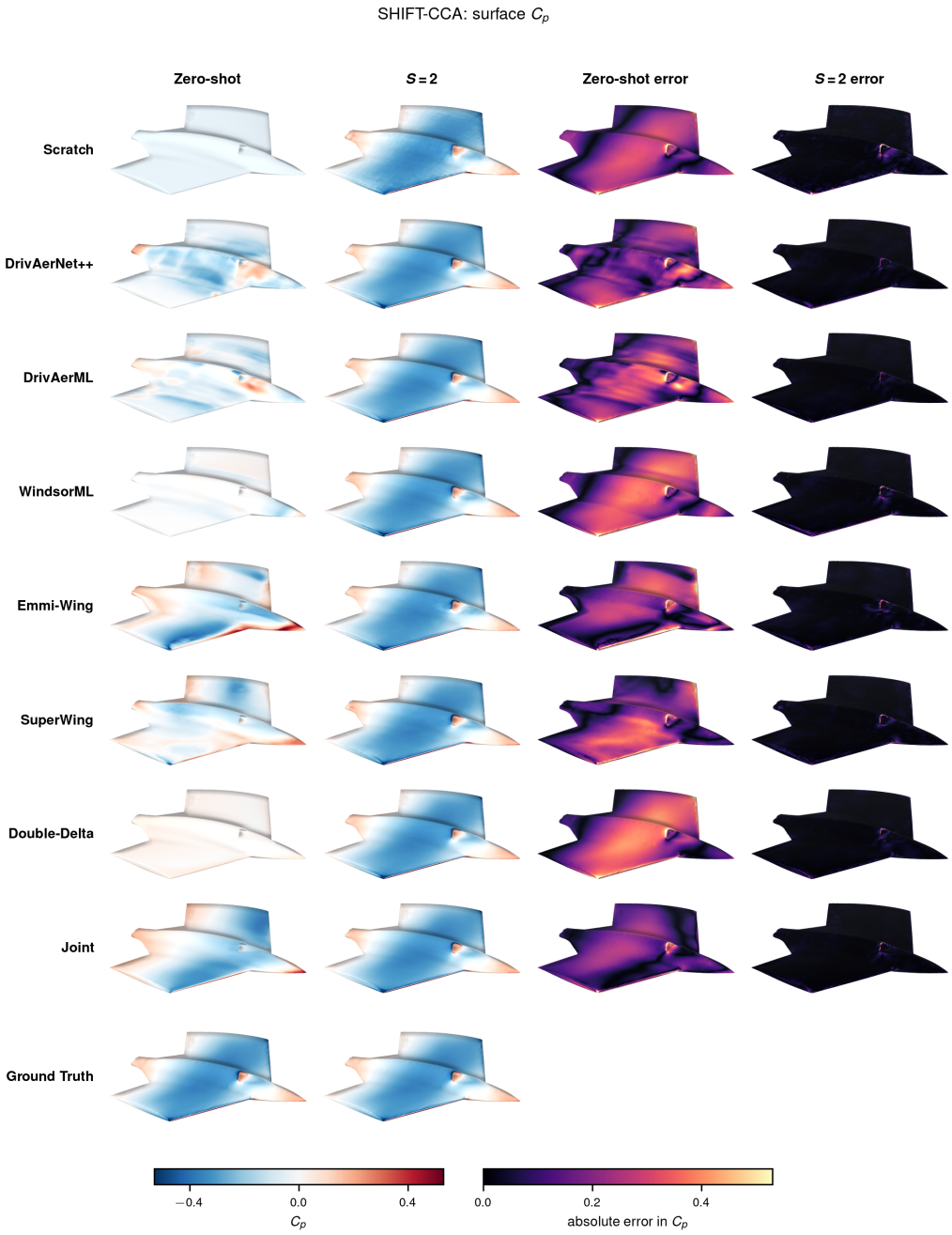

Figure 18: Model predictions of *surface* C_p on SHIFT-CCA.

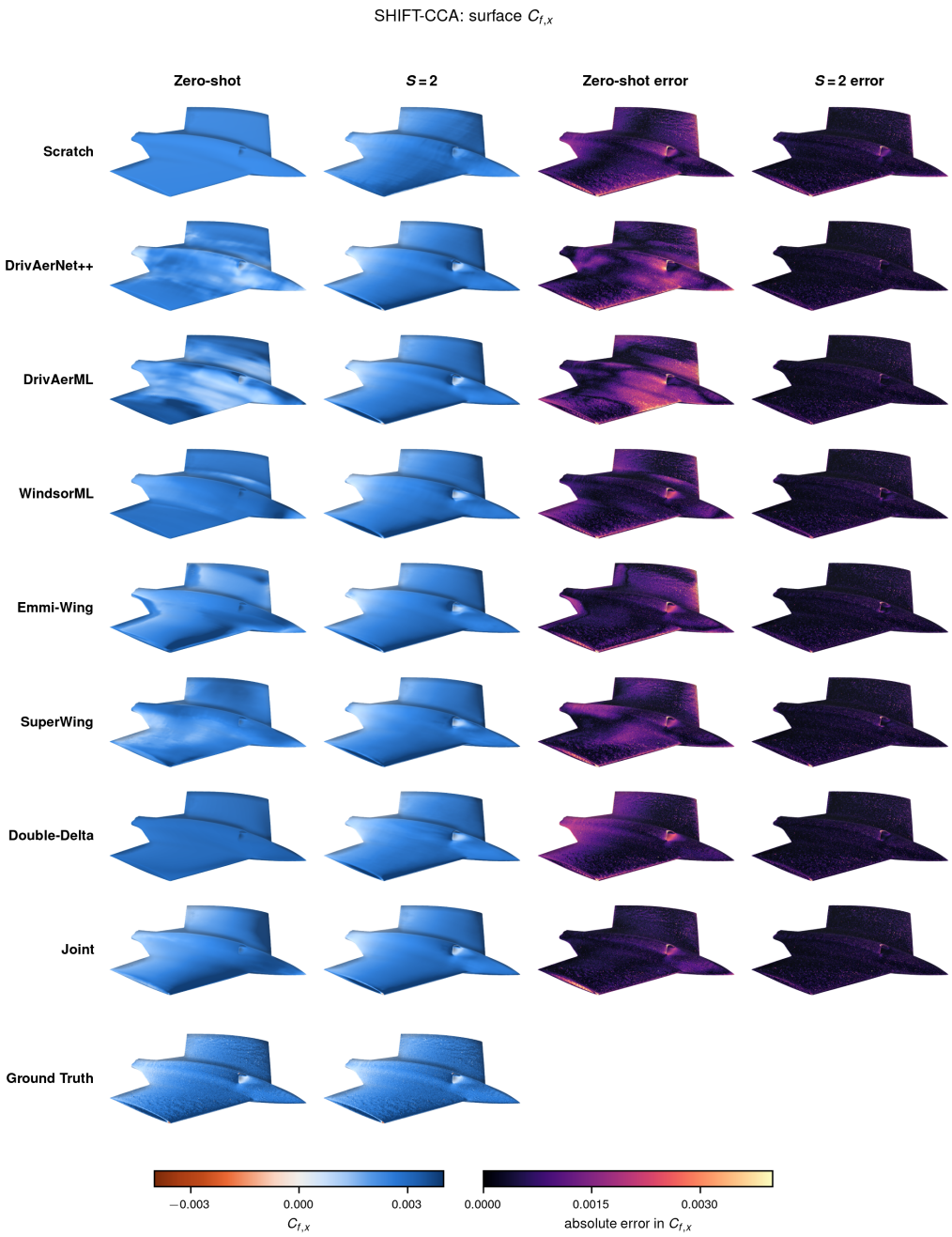

Figure 19: Model predictions of *surface* C_{f_x} on SHIFT-CCA.

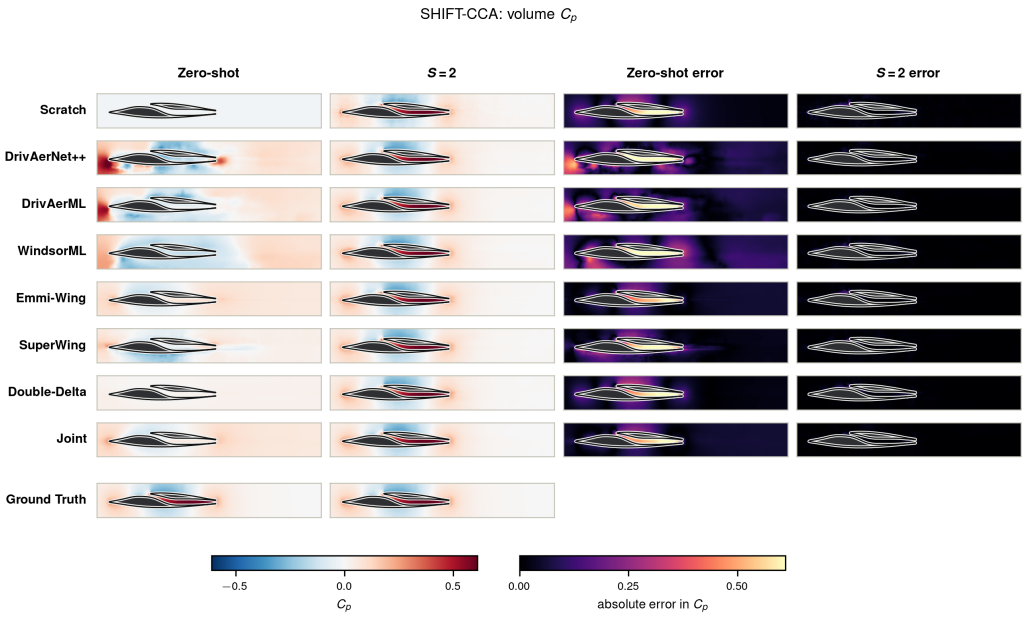

Figure 20: Model predictions of *volume* C_p on SHIFT-CCA.

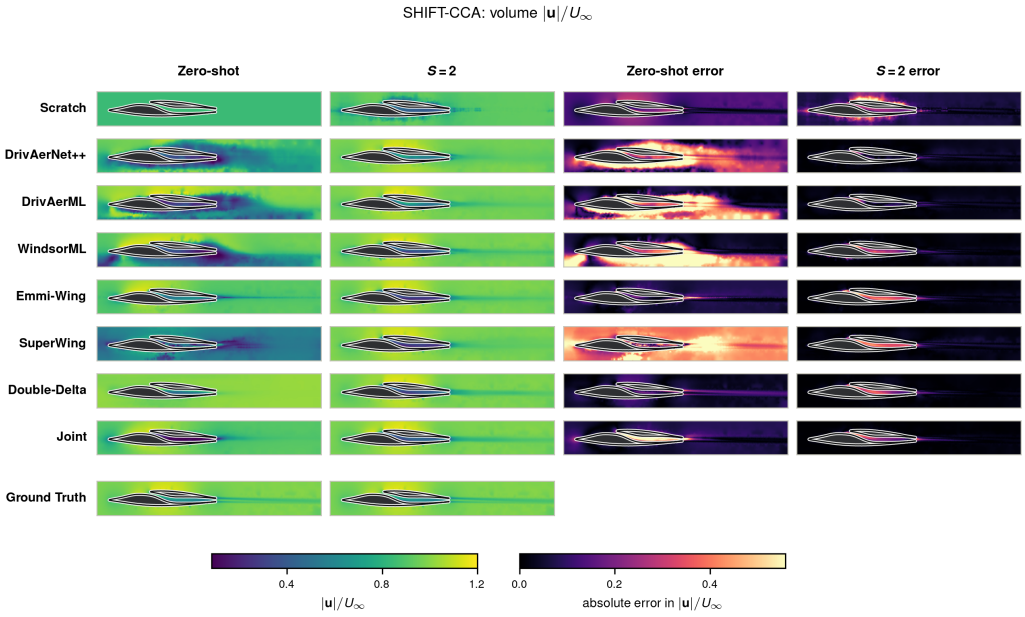

Figure 21: Model predictions of *volume* $|\mathbf{u}^*|$ on SHIFT-CCA

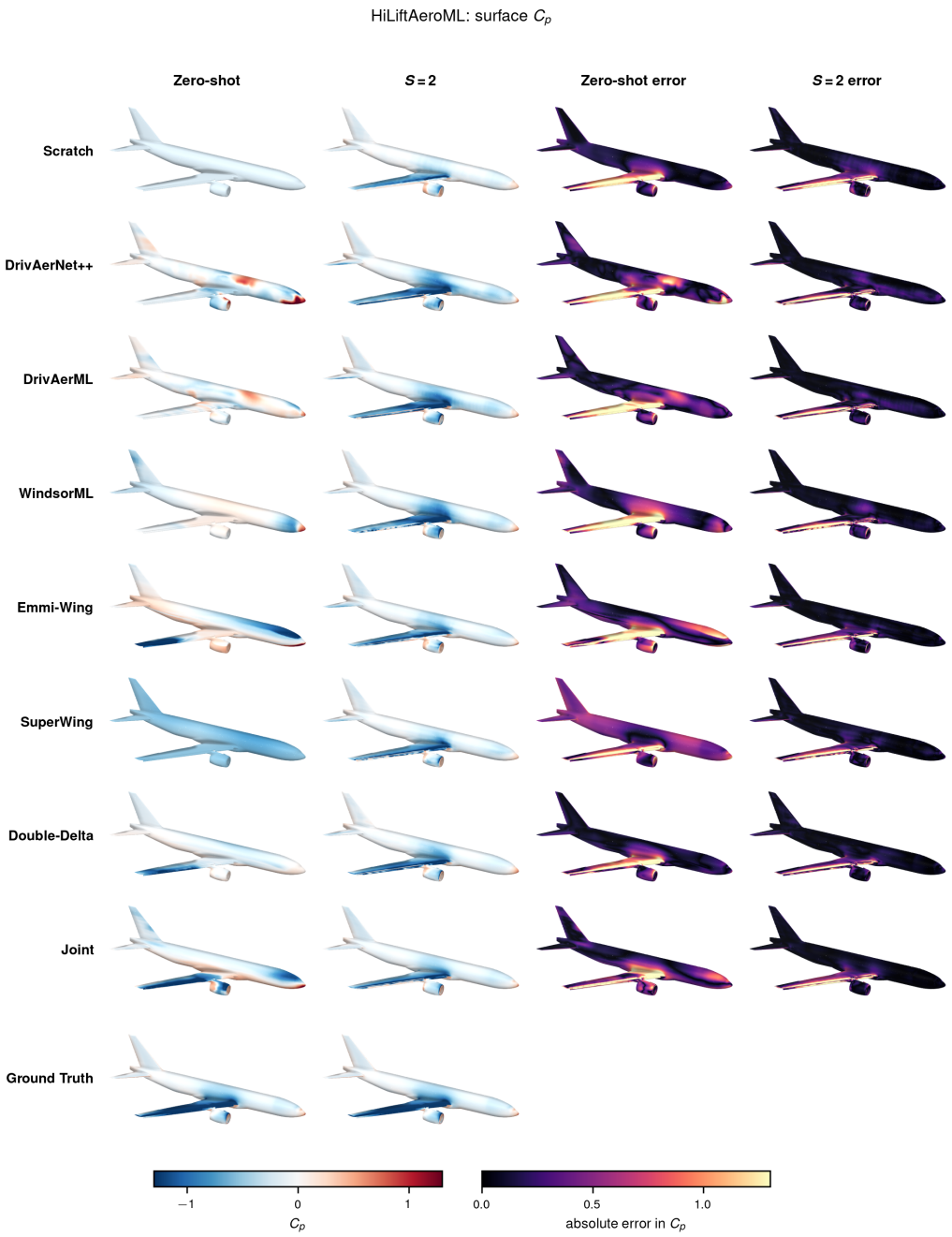

Figure 22: Model predictions of *surface* C_p on HiLiftAeroML.

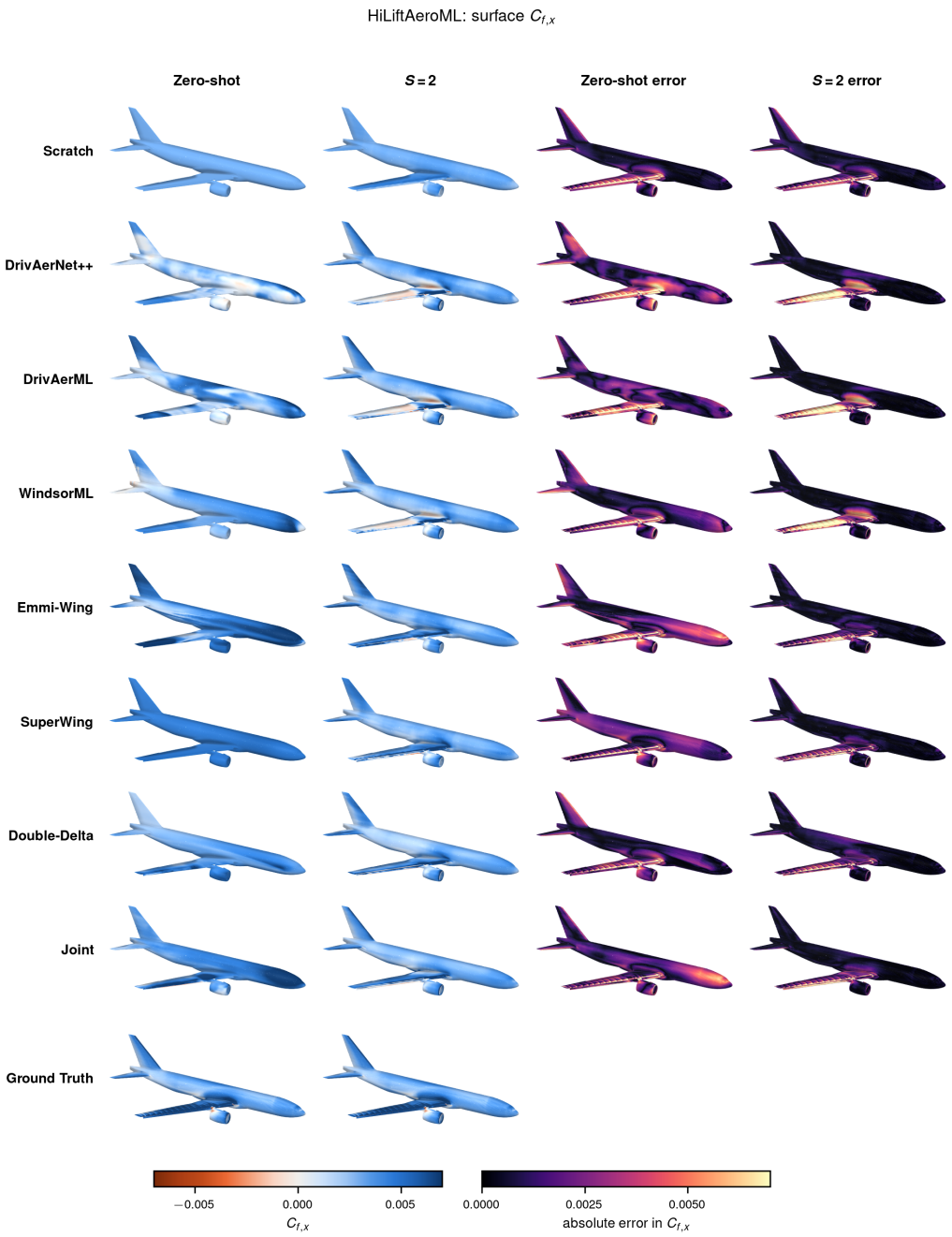

Figure 23: Model predictions of *surface* C_{f_x} on HiLiftAeroML.

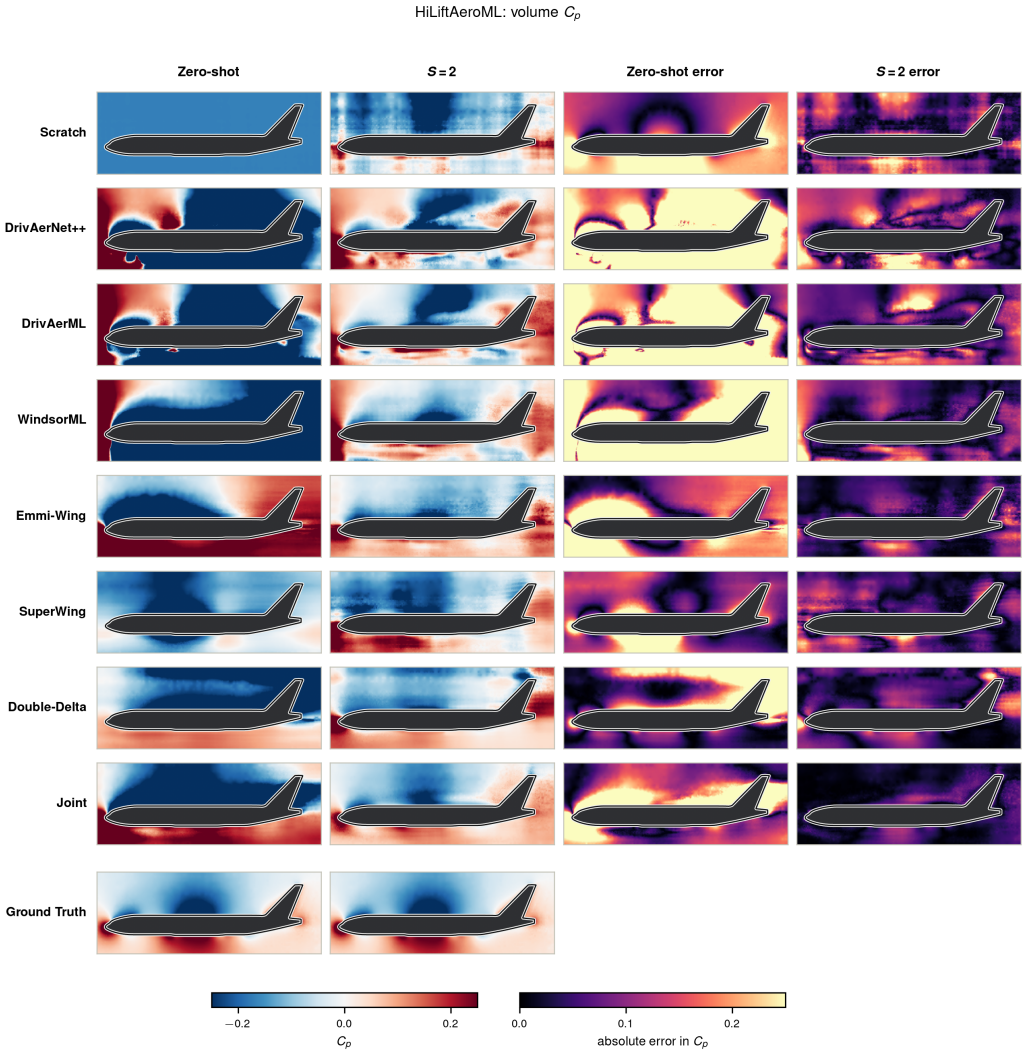

Figure 24: Model predictions of *volume* C_p on HiLiftAeroML.

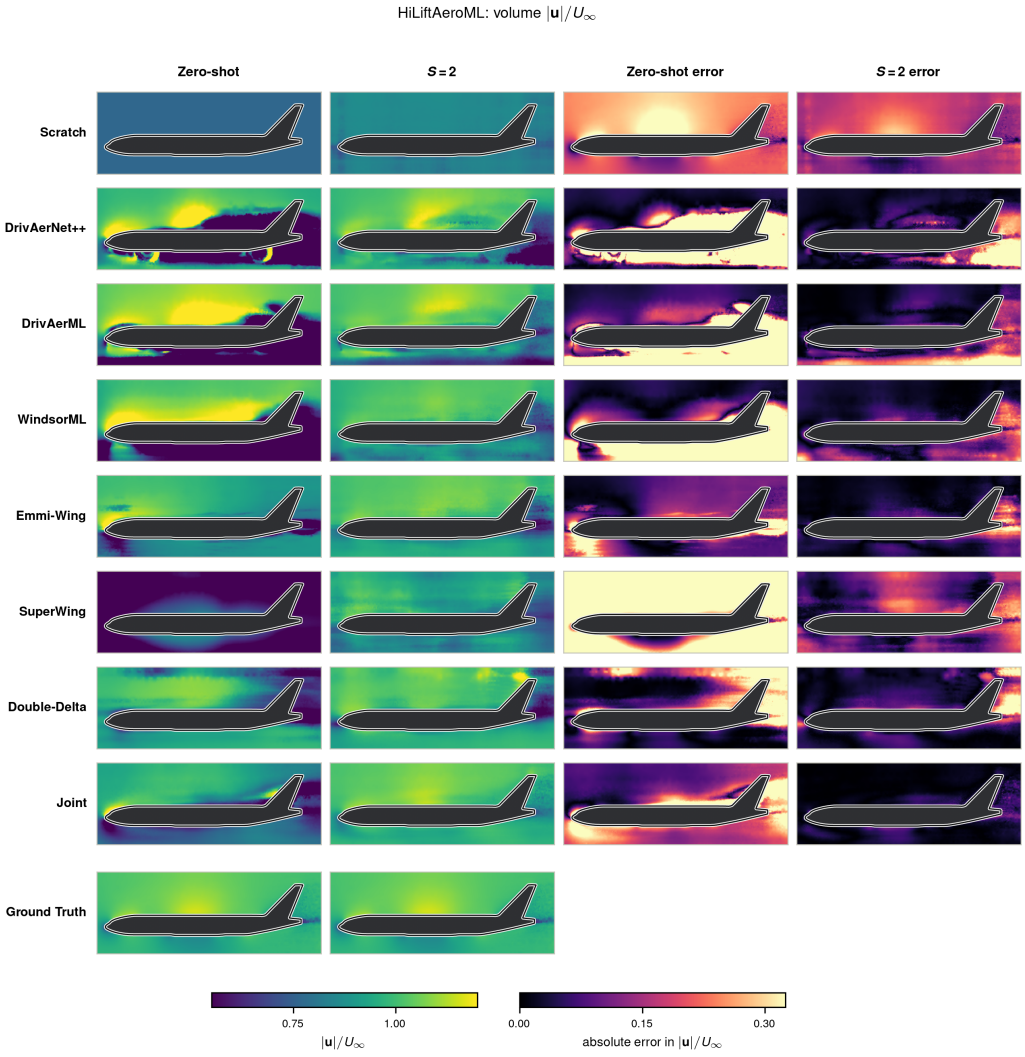

Figure 25: Model predictions of *volume* $|\mathbf{u}^*|$ on HiLiftAeroML