ADASOLVER: EFFICIENT NEURAL OPERATOR INFERENCE VIA TEST-TIME ADAPTIVE SAMPLING

Anonymous authors
Paper under double-blind review

ABSTRACT

Neural operators provide efficient surrogates for partial differential equations (PDEs), yet their inference performance is typically tied to uniform spatial discretizations or sampling schemes. Many PDE solutions exhibit localized structures that require dense sampling for accurate prediction, while smooth regions can be represented with substantially fewer points. Naively changing the sampling density, however, shifts the empirical input measure and can introduce sampling-dependent bias into the learned neural operator. We introduce **AdaSolver**, a measure-aware framework for adaptive spatial computation at test time. We characterize sampling-induced bias and its propagation through physics-attention, and derive a sampling-density-consistent Transolver using self-normalized importance weighting. The resulting measure-consistent formulation preserves consistency with the reference measure, enabling adaptive test-time sampling across different physical systems and geometric designs. Building on this formulation, we explore three refinement strategies based on physics-informed priors, coarse-prediction guidance, and cache-aware learned refinement. Across five standard PDE benchmarks and three industrial aerodynamic tasks, AdaSolver improves prediction accuracy under matched sampling budgets and accelerates surrogate-based airfoil and wing optimization. These results establish adaptive spatial sampling as a principled axis of test-time scaling for neural operators.

1 INTRODUCTION

Neural operators learn maps between function spaces and can amortize expensive numerical simulation across many physical configurations (Li et al., 2021; 2023b; Wu et al., 2024). Recently, transformer-based neural operators have improved the scalability and accuracy of operator learning across diverse PDEs and million-point geometries. They make such large-scale computation possible by compressing irregular meshes into latent tokens, distributing the computation, or decoupling global state estimation from pointwise decoding (Alkin et al., 2024; Luo et al., 2025; Zhou et al., 2026). In engineering applications, however, evaluation efficiency is as important as scalability. Yet how neural operators can flexibly allocate computation at inference remains largely unexplored.

Neural-operator training data are typically generated by numerical solvers on meshes that are uniform or designed for numerical convergence. Learned models are consequently evaluated under the same sampling distribution, which may ignore the strongly localized structure of many PDE solutions. Shocks, boundary layers, contact interfaces, and stress concentrations often occur in localized regions and require finer resolution, whereas smooth regions can be represented with fewer samples. This creates a dilemma: preserving the training distribution wastes samples in smooth regions, while reallocating them to critical regions changes the sampling measure. Can a neural operator reallocate a given sampling budget without changing the operator it represents?

Existing efforts address neighboring capabilities. First, resolution generalization studies how neural operators extrapolate across grid sizes, point counts, or compatible discretizations while preserving the underlying sampling distribution (Li et al., 2021; 2023b; Liu et al., 2023; Alkin et al., 2024; Zhou et al., 2026; Gao et al., 2025). These methods make resolution changes feasible, but do not by themselves provide a mechanism for spatially reallocating samples according to local physical or solution structures. Second, adaptive sampling and refinement methods predict where a numerical mesh or point set should be refined, often as part of a solver pipeline or through a separately trained

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.

refinement model (Huang et al., 2021; Obiols-Sales et al., 2023; Rowbottom et al., 2024). They address spatial allocation, but do not specify how a frozen neural operator should preserve its target integral when that allocation changes the sampling measure. Third, test-time compute scaling increases inference effort through adaptive reasoning or iterative physical prediction (Snell et al., 2024; Lippe et al., 2023; Jenkins, 2026; Serrano et al., 2026). These works establish the value of allocating additional computation at inference, but leave open the spatial counterpart: reallocating input samples for operator evaluation while retaining the same learned map.

We introduce AdaSolver to address this gap. We view spatial sampling density as a controllable inference resource for allocating computation at test time. From a continuous-operator perspective, we characterize how shifts in sampling density introduce sampling bias into Transolver. We develop a sampling-density-consistent Transolver that corrects this bias through self-normalized importance weighting. This measure correction enables adaptive test-time sampling driven by geometry or physics priors, a low-cost coarse prediction, or a cache-aware learned refinement policy that reuses intermediate features. Figure 1 illustrates the AdaSolver framework and its alternative test-time sampling strategies.

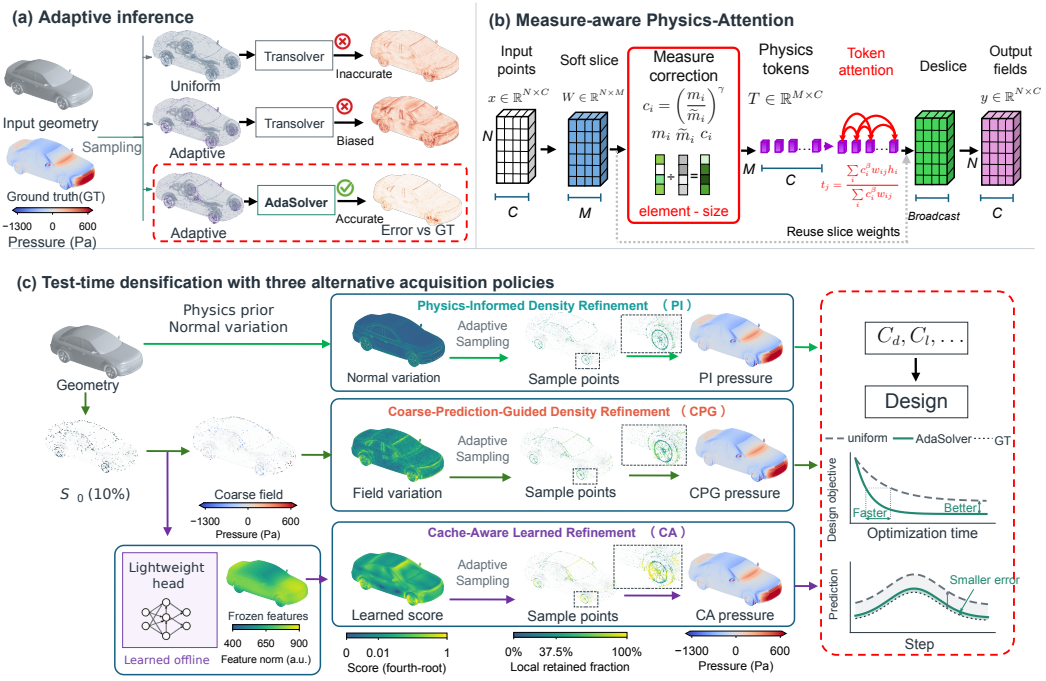

Figure 1: **AdaSolver overview.** (a) Adaptive inference with measure correction. (b) Corrected aggregation, token attention, and deslicing. (c) Geometry priors, coarse predictions, and a lightweight head on frozen features provide alternative refinement signals. The frozen operator evaluates the selected points and supplies aerodynamic quantities for design. The rightmost curves illustrate the design objectives schematically.

We summarize our contributions as follows: (i) to our knowledge, we provide the first systematic exploration of adaptive spatial sampling as a new axis of test-time scaling for neural operators, revealing sampling-measure shift as a fundamental challenge when reallocating spatial computation at inference; (ii) we introduce AdaSolver, which enables measure-consistent test-time sampling and supports diverse scaling strategies based on physics and geometry priors, coarse predictions, and learned refinement signals; and (iii) we demonstrate consistent improvements in the accuracy–compute trade-off across five standard PDE benchmarks and three industrial aerodynamic tasks, and further show that AdaSolver accelerates surrogate-based airfoil and wing optimization while achieving comparable or improved CFD-validated aerodynamic performance.

2 Related Work

Neural operators on irregular geometries. Fourier neural operators learn resolution-transferable spectral mappings on regular grids, while Geo-FNO extends them to general geometries through learned coordinate deformations (Li et al., 2021; 2023a). GINO couples integral operators with a latent regular grid, and NUNO partitions non-uniform observations to reduce interpolation and computational costs (Li et al., 2023b; Liu et al., 2023). Transformer-based operators instead compress large point sets into latent representations before global interaction, as in GNOT, UPT, and Transolver (Hao et al., 2023; Alkin et al., 2024; Wu et al., 2024). Transolver++ and Transolver-3 further scale this design to million-scale and industrial-scale geometries (Luo et al., 2025; Zhou et al., 2026). These methods establish strong compatibility with irregular inputs. Our focus is complementary: we actively change the *local* sampling density at inference and ask whether the model can preserve operator consistency while maintaining or even improving predictive performance.

Discretization-consistent operator learning. Extending finite neural architectures to function spaces requires discrete reductions to converge to fixed continuous integrals as the discretization changes (Berner et al., 2026). Monte Carlo convolution corrects non-uniform point-cloud sampling with inverse-density weights, while QuadConv evaluates continuous kernels using quadrature on irregular PDE data (Hermosilla et al., 2018; Doherty et al., 2024). Related formulations construct discretization-invariant maps between neural fields and quadrature-aware normalization statistics (Wang & Golland, 2022; Kim et al., 2026). Nevertheless, resolution-independent parameters alone do not prevent errors under a mismatch between training and evaluation discretizations (Gao et al., 2025). We adopt the same function-space principle for latent point aggregation, but use it to support a deliberate inference-time change of sampling density rather than only passive resolution transfer.

Adaptive meshes and test-time refinement. Adaptive discretization concentrates resolution where numerical error is expected to be large, and learned methods predict mesh refinement or node relocation for numerical solvers (Huang et al., 2021; Obiols-Sales et al., 2023; Rowbottom et al., 2024). Test-time compute has also been increased through iterative refinement, adaptive tokenization, and neural-operator composition (Lippe et al., 2023; Jenkins, 2026; Wang & Perdikaris, 2026; Serrano et al., 2026). These methods determine where or how inference computation is allocated. AdaSolver instead addresses the operator-level challenge of changing spatial sampling density while preserving the reference operator.

3 Method

3.1 Transolver as an Integral Operator

Transolver (Wu et al., 2024) approximates continuous integral operators through Physics-Attention. Let Ω denote the physical domain, and let ρ denote the sampling measure induced by the chosen grid or mesh discretization policy. A discretized example consists of pointwise samples $\{(x_i, a_i)\}_{i=1}^N \mapsto \{u_i\}_{i=1}^N$, where $x_i \sim \rho$, $a_i = a(x_i)$ is the input field, and $u_i = u(x_i)$ is the output field. A neural operator learns an operator $\mathcal{G} : a \mapsto u$ with a parameterized network $\mathcal{G}_\theta$.

Physics-Attention projects point features into M slice weights, applies attention to the resulting latent physical states, and broadcasts the updated states back to the points. Let $h_i \in \mathbb{R}^C$ denote the C-dimensional feature embedding of (x_i, a_i). The slice weights are

$$w_i = \mathrm{Softmax}\left(\frac{\mathrm{Linear}(h_i)}{\tau_0}\right) \in \mathbb{R}^M, \qquad \sum_{j=1}^{M} w_{i,j} = 1, \tag{1}$$

where τ_0 is the temperature. The corresponding physics token is a normalized weighted average

$$u_{s,j}^{(N)} = \frac{\frac{1}{N}\sum_{i=1}^N w_{i,j} h_i}{\frac{1}{N}\sum_{i=1}^N w_{i,j}} \xrightarrow[N\to\infty]{} u_{s,j}^{\rho} = \frac{\int_\Omega w_j(x) h(x)\,\mathrm{d}\rho(x)}{\int_\Omega w_j(x)\,\mathrm{d}\rho(x)}. \tag{2}$$

Writing $u_s^{(N)} = [u_{s,1}^{(N)}, \dots, u_{s,M}^{(N)}]^\top \in \mathbb{R}^{M\times C}$ and $q, k, v = \mathrm{Linear}(u_s^{(N)})$, the discrete state attention and its continuous integral counterpart can be written together as

$$\begin{aligned} u'_{s,j} &= \sum_{t=1}^{M} \frac{\exp\big(q_j^\top k_t/\tau\big)}{\sum_{p=1}^{M}\exp\big(q_j^\top k_p/\tau\big)} v_t \xrightarrow[M\to\infty]{} \mathcal{A}[u_s^\rho](\xi'_s), \\ \mathcal{A}[u_s^\rho](\xi'_s) &:= \int_{\Omega_s} \frac{\exp\big(q(u_s^\rho(\xi'_s))^\top k(u_s^\rho(\xi_s))/\tau\big)}{\int_{\Omega_s}\exp(q(u_s^\rho(\xi'_s))^\top k(u_s^\rho(\eta_s))/\tau)\,\mathrm{d}\eta_s} v(u_s^\rho(\xi_s))\,\mathrm{d}\xi_s. \end{aligned} \tag{3}$$

where $u_s^\rho(\xi_s)$ is the continuous physics-token field on the slice domain Ω_s, ξ'_s is a query location, and τ is the attention temperature. Finally, the updated tokens are mapped back to the point space through the slice weights,

$$h'_i = \sum_{j=1}^{M} w_{i,j} u'_{s,j}, \qquad i = 1, \dots, N. \tag{4}$$

Thus, Physics-Attention replaces the $N \times N$ point interaction with operations of complexity $\mathcal{O}(NMC + M^2C)$ for $M \ll N$. During training, the input samples used by $\mathcal{G}_\theta$ are typically generated according to the prescribed measure ρ.

3.2 Sampling-Density-Consistent Transolver

Transolver is trained on point sets drawn from a prescribed sampling measure ρ, induced by the discretization used to generate the training data, such as a uniform grid or a refined mesh. At inference time, we may instead sample from a different measure $\widetilde{\rho}$ to obtain a different spatial description of the same physical domain. For example, aerodynamic geometries may be sampled more densely near surfaces or flow-sensitive regions and more sparsely in smooth regions. However, changing the sampling measure alters the Monte Carlo aggregation in Transolver and therefore changes its limiting operator.

Let $\lambda = \mathrm{d}\widetilde{\rho}/\mathrm{d}\rho > 0$ denote the density ratio. For a slice location $\xi_s \in \Omega_s$, define the slice-weighted probability measure

$$\mathrm{d}\pi_{\xi_s}(x) := \frac{w(x,\xi_s)}{\int_\Omega w(y,\xi_s)\,\mathrm{d}\rho(y)}\,\mathrm{d}\rho(x). \tag{5}$$

Proposition 3.1 (Physics-token bias under a sampling-density shift)**.** *For each $\xi_s \in \Omega_s$, the reference and shifted physics tokens satisfy*

$$u_s^{\widetilde{\rho}}(\xi_s) - u_s^\rho(\xi_s) = \frac{\mathrm{Cov}_{\pi_{\xi_s}}(\lambda, h)}{\mathbb{E}_{\pi_{\xi_s}}[\lambda]}. \tag{6}$$

Moreover, if $\mathcal{P}_{\xi_s}^\mu h := u_s^\mu(\xi_s)$ denotes the tokenization map under a sampling measure μ, then

$$\left\| \mathcal{P}_{\xi_s}^{\widetilde{\rho}} - \mathcal{P}_{\xi_s}^{\rho} \right\|_{\mathrm{op}} = \frac{\sqrt{\mathrm{Var}_{\pi_{\xi_s}}(\lambda)}}{\mathbb{E}_{\pi_{\xi_s}}[\lambda]}. \tag{7}$$

Let $\mathcal{T}$ denote the continuous Physics-Attention operator and define $\mathcal{G}_\mu(h) := \mathcal{T}[u_s^\mu]$. Write $\delta_s := u_s^{\widetilde{\rho}} - u_s^\rho$ and let $R_{\delta_s} := \alpha_{u_s^{\widetilde{\rho}}}/\alpha_{u_s^\rho}$ be the ratio between the shifted and reference attention kernels.

Theorem 3.2 (Propagation through continuous Physics-Attention)**.** *The sampling-density bias propagates to the Physics-Attention output as*

$$\begin{aligned} \mathcal{G}_{\widetilde{\rho}}(h)(x) - \mathcal{G}_\rho(h)(x) = \int_{\Omega_s} w(x,\xi'_s) \int_{\Omega_s} \alpha_{u_s^\rho}(\xi'_s,\xi_s) \Big[&(R_{\delta_s}(\xi'_s,\xi_s) - 1) W_v u_s^\rho(\xi_s) \\ &+ R_{\delta_s}(\xi'_s,\xi_s) W_v \delta_s(\xi_s) \Big] J_g(\xi_s)\,\mathrm{d}\xi_s\,\mathrm{d}\xi'_s. \end{aligned} \tag{8}$$

Here $\alpha_{u_s^\rho}$ is the reference attention kernel, W_v is the value projection, and J_g is the physical-to-slice Jacobian. The two terms respectively describe the redistribution of attention weights and the propagation of shifted value tokens.

The complete conditions and proofs of Proposition 3.1 and Theorem 3.2 are given in Appendix A.

These results identify the point-to-slice aggregation as the source of the sampling-density bias. The slice assignment $w_{i,j}$ is computed independently at each point, attention is subsequently applied among a fixed number of physics tokens, and deslicing maps the updated tokens back to each point. Only the physics-token construction aggregates over the sampled points. Consequently, for $x_i \sim \widetilde{\rho}$, the original aggregation converges to

$$\frac{\sum_{i=1}^{N} w_{i,j} h_i}{\sum_{i=1}^{N} w_{i,j}} \xrightarrow[N\to\infty]{} \frac{\int_\Omega w_j(x) h(x)\, \mathrm{d}\widetilde{\rho}(x)}{\int_\Omega w_j(x)\, \mathrm{d}\widetilde{\rho}(x)} = u_{s,j}^{\widetilde{\rho}}, \tag{9}$$

rather than the reference token $u_{s,j}^{\rho}$ learned under ρ. We therefore correct the measure only when forming the physics tokens. Let the measure-correction weight be

$$c_i := \frac{\mathrm{d}\rho}{\mathrm{d}\widetilde{\rho}}(x_i) = \lambda(x_i)^{-1}, \tag{10}$$

which corresponds to the inverse relative inclusion density under discrete sampling. The corrected aggregation becomes

$$\widehat{u}_{s,j}^{(N)} := \frac{\sum_{i=1}^{N} c_i w_{i,j} h_i}{\sum_{i=1}^{N} c_i w_{i,j}} \xrightarrow[N\to\infty]{} \frac{\int_\Omega w_j(x) h(x)\, \mathrm{d}\rho(x)}{\int_\Omega w_j(x)\, \mathrm{d}\rho(x)} = u_{s,j}^{\rho}. \tag{11}$$

Because the estimator is self-normalized, only relative density ratios are required. After the corrected physics tokens are obtained, we apply the original attention among physics tokens and the original deslice operation without further modification. For a Transolver with L layers, the same correction is applied whenever layer ℓ forms new physics tokens from its current point features $h_i^{(\ell)}$:

$$\widehat{u}_{s,j}^{(\ell)} = \frac{\sum_{i=1}^{N} c_i w_{i,j}^{(\ell)} h_i^{(\ell)}}{\sum_{i=1}^{N} c_i w_{i,j}^{(\ell)}}, \qquad \ell = 1, \dots, L. \tag{12}$$

Thus, for any admissible sampling density $\widetilde{\rho}$, all learned projections and attention parameters remain unchanged, and only the point-to-slice aggregation is reweighted.

3.3 Adaptive Test-Time Sampling for Transolver

We next construct the inference-time sampling measure from geometry, a draft prediction, or intermediate backbone features.

Let $r_\psi : \Omega \to [0, \infty)$ be a nonnegative refinement score, where ψ denotes the information used to construct it, and define

$$\nu_\psi(x) := \frac{r_\psi(x)}{\int_\Omega r_\psi(y)\, \mathrm{d}\rho(y)}, \qquad \widetilde{\rho}_{\gamma,\psi} := (1-\gamma)\rho + \gamma(\nu_\psi * \rho), \qquad \gamma \in [0,1]. \tag{13}$$

Here $\nu_\psi * \rho$ denotes the reweighted empirical measure induced by the refinement score. The correction used in the point-to-slice aggregation is

$$c_{\gamma,\psi}(x) := \frac{\mathrm{d}\rho}{\mathrm{d}\widetilde{\rho}_{\gamma,\psi}}(x) = \frac{1}{(1-\gamma) + \gamma \nu_\psi(x)}. \tag{14}$$

We further parameterize the strength of measure correction as

$$c_{\gamma,\psi}^{(\beta)}(x) := c_{\gamma,\psi}(x)^\beta, \qquad \beta \in [0,1], \tag{15}$$

where β controls the strength of density compensation: $\beta = 0$ recovers the original Transolver without correction, while larger β increases the inverse-density reweighting. We use a fixed policy-specific β in the main experiments and ablate it in Appendix D.1 with the sampled points held fixed.

3.3.1 PHYSICS-INFORMED DENSITY REFINEMENT

The first policy constructs r_ψ from prior information about the geometry or the physical problem. Let $s_{\mathrm{prior}}(x) \geq 0$ be a problem-dependent indicator whose large values identify regions expected to require finer resolution. This indicator can be obtained from a known geometric feature, a numerical refinement rule, or a physical quantity available before inference. We set

$$\nu_{\mathrm{prior}}(x) := \frac{s_{\mathrm{prior}}(x)}{\int_\Omega s_{\mathrm{prior}}(y)\,\mathrm{d}\rho(y)} \tag{16}$$

and use it in equation 13 to draw the inference samples. This policy does not require a preliminary network evaluation; once the prior indicator is available, points are sampled from $\widetilde{\rho}_{\gamma,\mathrm{prior}}$ and inferred with the correction in equation 14.

3.3.2 COARSE-PREDICTION-GUIDED DENSITY REFINEMENT

This strategy is motivated by PDE-Refiner (Lippe et al., 2023) and ATCA (Jenkins, 2026). When no reliable prior is available, we draw a small coarse sample S_0 from ρ and evaluate the frozen operator:

$$\widehat{u}^{(0)} := \mathcal{G}_\theta\Big(a\big|_{S_0}\Big), \qquad |S_0| \ll N, \tag{17}$$

where $a\big|_{S_0}$ denotes the input field restricted to S_0. We then construct a nonnegative refinement score from the coarse prediction, for example,

$$s_{\mathrm{coarse}}(x) := \Phi\Big(\widehat{u}^{(0)}(x), \nabla\widehat{u}^{(0)}(x), \Delta\widehat{u}^{(0)}(x), \dots\Big), \qquad \Phi \geq 0, \tag{18}$$

where Φ denotes a task-dependent combination of coarse predictive features. These quantities are estimated from the local discrete neighborhood and extended to candidate locations when necessary. The resulting score is normalized through equation 13 to define $\widetilde{\rho}_{\gamma,\mathrm{coarse}}$. The resulting density concentrates samples in regions that the coarse prediction identifies as spatially informative or task-relevant.

3.3.3 CACHE-AWARE LEARNED REFINEMENT

Coarse-prediction-guided refinement requires a complete forward pass on S_0 solely to estimate the sampling density. Moreover, once new samples are introduced, the resulting change in the global physics tokens prevents most of this coarse computation from being reused. We address both limitations with a learned early refinement signal and a cache-aware final forward pass. Keeping the backbone frozen, we train a small refinement head $H_\phi : \mathbb{R}^C \to \mathbb{R}_+$ on the first-block features $h_i^{(1)}$ to directly predict the refinement score,

$$r_{\mathrm{CA}}(x_i) := H_\phi\Big(h_i^{(1)}\Big). \tag{19}$$

The predicted score is then used in equation 13 to construct the adaptive sampling density, avoiding a complete coarse forward pass.

Let $S_1 = S_0 \cup \Delta S$, where ΔS contains the newly sampled points. For the first Physics-Attention layer, the pointwise input features h_i^{in} and slice weights $w_{i,j}^{(1)}$ computed on S_0 remain unchanged when ΔS is added and can therefore be cached. After the final sampling density and measure-correction weights c_i are determined, the first physics token is formed as

$$\widehat{u}_{s,j}^{(1)} = \frac{\sum_{i\in S_0} c_i w_{i,j}^{(1)} h_i^{\mathrm{in}} + \sum_{i\in\Delta S} c_i w_{i,j}^{(1)} h_i^{\mathrm{in}}}{\sum_{i\in S_0} c_i w_{i,j}^{(1)} + \sum_{i\in\Delta S} c_i w_{i,j}^{(1)}}. \tag{20}$$

Thus, the pointwise quantities for S_0 are reused, while only those for ΔS need to be newly computed. Token attention, deslicing, and later blocks are recomputed on S_1, since the added samples alter the global physics tokens. Appendix C provides the task-specific density constructions.

4 EXPERIMENTS

We evaluate AdaSolver on five standard PDE benchmarks and three industrial design prediction tasks. We further assess its efficiency in two CFD-validated geometry optimization tasks involving two-dimensional airfoils and three-dimensional wings.

4.1 BENCHMARKS

Our evaluation spans canonical PDE problems and representative industrial applications. The standard PDE suite comprises Elasticity, Plasticity, Airfoil, Pipe (Li et al., 2023a) and Darcy flow (Li et al., 2021). These public datasets are widely used to benchmark neural operators (Hao et al., 2023; Wu et al., 2024). For industrial applications, we consider ShapeNet-Car (Umetani & Bickel, 2018) and DrivAerML (Ashton et al., 2024) for automotive aerodynamics, and SuperWing (Yang et al., 2025) for three-dimensional wing design. In addition to physical-field prediction errors, we evaluate the accuracy of design-relevant aerodynamic quantities, including the drag coefficient C_D, lift coefficient C_L, and drag-to-lift ratio C_D/C_L. We also use the learned PDE surrogates to guide shape optimization of a two-dimensional airfoil and a three-dimensional wing, assessing computational efficiency and aerodynamic prediction accuracy during optimization. Governing equations, data splits, and input–output mappings are detailed in Appendix B.

4.2 IMPLEMENTATION AND EVALUATION

As described in equation 2 and equation 3, Transolver approximates a continuous integral operator through point aggregation and attention over latent physical states. With the number of physical states fixed, varying the spatial sampling budget changes the number of point samples used to estimate the aggregation integrals. For Airfoil, Pipe, Elasticity, and Plasticity, we adopt geometry-amortized training from Transolver-3 (Zhou et al., 2026). Given a full mesh of N points, we draw a point count N_t uniformly from $\{\mathrm{round}(0.4N), \ldots, N\}$ for each minibatch, then sample N_t mesh points uniformly without replacement. Darcy uses the existing backbone trained on full meshes. The trained backbone is frozen when evaluating different sampling policies and inference budgets.

For a consistent comparison, we treat all input discretizations as unstructured meshes and evaluate sampling budgets of 37.5%, 50%, and 75% of the full-mesh point count. Uniform subsampling of mesh points serves as the common baseline. We report relative L_2 errors for physical-field predictions and summarize performance across budgets using the relative area under the error–budget curve (rAUEC). Let b_k denote the retained fraction of mesh points, with $(b_1, b_2, b_3) = (0.375, 0.50, 0.75)$, and let $E_{\mathrm{method}}(b)$ and $E_{\mathrm{uniform}}(b)$ denote mean test errors for a given sampling seed. Using trapezoidal integration, we compute

$$\mathrm{rAUEC} := \frac{\sum_{k=1}^{2}(b_{k+1}-b_k)\big[E_{\mathrm{method}}(b_k)+E_{\mathrm{method}}(b_{k+1})\big]}{\sum_{k=1}^{2}(b_{k+1}-b_k)\big[E_{\mathrm{uniform}}(b_k)+E_{\mathrm{uniform}}(b_{k+1})\big]}. \tag{21}$$

Lower rAUEC is better, with **uniform sampling normalized to** 1.

4.3 MAIN RESULTS

Standard Benchmarks. Table 1 compares physics-informed density refinement (PI), coarse-prediction-guided density refinement (CPG), and cache-aware learned refinement (CA) against uniform subsampling across five standard PDE benchmarks. Overall, adaptive test-time sampling improves the error–budget trade-off, demonstrating that sampling density provides a controllable axis for test-time computation. CPG generally outperforms PI, suggesting that solution-dependent variation can provide more informative refinement cues than geometry or prescribed heuristics alone. CA performs comparably to CPG on several tasks, showing that such solution-aware signals can be recovered from intermediate features without requiring a complete coarse prediction. The distinction is particularly clear on Darcy: despite its regular Cartesian grid, PI offers little improvement over uniform sampling, whereas CPG and CA substantially reduce rAUEC. This result indicates that the benefit of adaptive sampling is not limited to geometric complexity, but also depends on how well the sampling density captures localized structure in the solution.

Industrial Applications. Accurate aerodynamic surrogate modeling requires both high-fidelity full-field prediction and reliable estimation of design-relevant quantities. Table 2 evaluates both aspects on ShapeNet-Car, DrivAerML, and SuperWing. Across all three datasets, adaptive sampling consistently improves the error–budget trade-off for both field predictions and aerodynamic metrics. Notably, several aerodynamic quantities match or even surpass full-mesh accuracy using only about one-third of the points, indicating that spatial resolution can be concentrated in physically informative regions

Table 1: rAUEC (↓) on standard PDE benchmarks. Means appear above standard deviations ($n = 3$). Best and second-best means are **bold** and underlined, respectively.

Method	Airfoil2D		Pipe	Darcy	Elasticity	Plasticity
	Field	C_D/C_L	Field	Field	Field	Field
PI	0.6166 ±0.0197	**0.2174** ±0.1688	0.8711 ±0.0061	0.9965 ±0.0177	0.7280 ±0.0083	**0.8888** ±0.0041
CPG	0.5831 ±0.0219	0.2379 ±0.1781	**0.6719** ±0.0098	0.6419 ±0.0049	**0.4681** ±0.0041	0.8890 ±0.0038
CA	**0.5579** ±0.0139	0.2453 ±0.1779	0.7051 ±0.0116	**0.6400** ±0.0062	0.5469 ±0.0021	0.8897 ±0.0049

without sacrificing global field fidelity. The qualitative results in Figure 2 support this interpretation: adaptive policies concentrate samples around localized solution structures and geometrically or physically sensitive regions, where the corresponding prediction residuals are also reduced relative to uniform sampling. Together, these results show that adaptive sampling improves accuracy by concentrating test-time resolution in more informative regions. Appendix D.1 isolates the effect of measure correction, followed by budget-wise error curves and full-field visualizations across all eight benchmarks.

Table 2: rAUEC (↓) on industrial benchmarks. Means appear above standard deviations ($n = 3$). Best and second-best means are **bold** and underlined, respectively.

Method	ShapeNet-Car		DrivAerML		SuperWing	
	Field	C_D	Field	C_D	Field	C_D/C_L
PI	0.58952 ±0.00180	0.65606 ±0.00168	0.88342 ±0.00011	0.92455 ±0.00149	0.75640 ±0.00079	0.82456 ±0.00539
CPG	**0.55770** ±0.00219	**0.64743** ±0.00454	**0.82458** ±0.00005	**0.92093** ±0.00148	**0.67059** ±0.00212	**0.80259** ±0.00761
CA	0.55774 ±0.00222	0.64786 ±0.00404	0.82635 ±0.00019	0.92438 ±0.00153	0.71927 ±0.00211	0.81675 ±0.00700

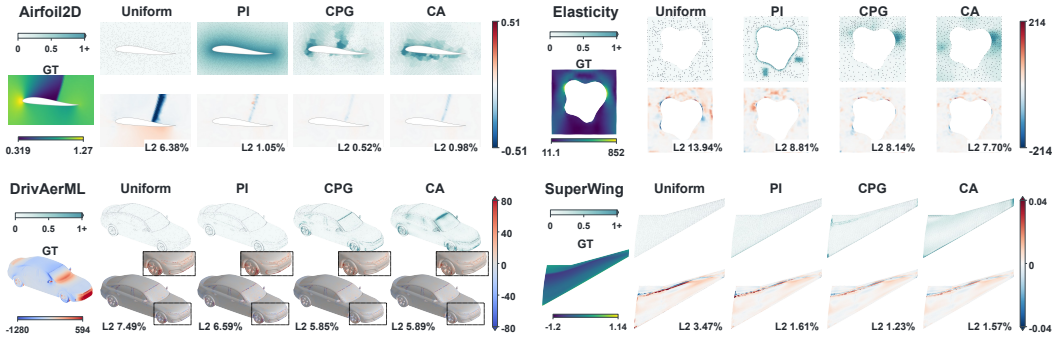

Figure 2: **Adaptive sampling and field prediction.** Four test cases at a 37.5% point budget. Upper rows show samples overlaid on sampling densities; lower rows show signed prediction errors relative to ground truth. Annotations report relative L_2 field errors.

4.4 Inverse Design

We evaluate AdaSolver in surrogate-based airfoil and wing design. The airfoil task minimizes $(C_D/C_L)^2$, while SuperWing minimizes drag subject to a fixed-lift target. Final designs are independently validated with CFD. Appendix B.5 details the geometry controls, optimization constraints, sampling schedules, and CFD settings.

2D Airfoil. We follow the airfoil inverse-design setting of Geo-FNO (Li et al., 2023a), parameterizing the geometry by spline-node displacements and performing gradient-based optimization through the differentiable surrogate. AdaSolver retains 2,113 of 11,271 points (18.75%) and keeps the

sampled indices fixed after initialization. It accelerates the complete design process by $2.18\times$ while reaching essentially the same CFD-validated aerodynamic performance as full-mesh optimization. It also yields a substantially more accurate surrogate estimate of the final design, demonstrating that adaptive test-time sampling can reduce optimization cost without compromising design quality.

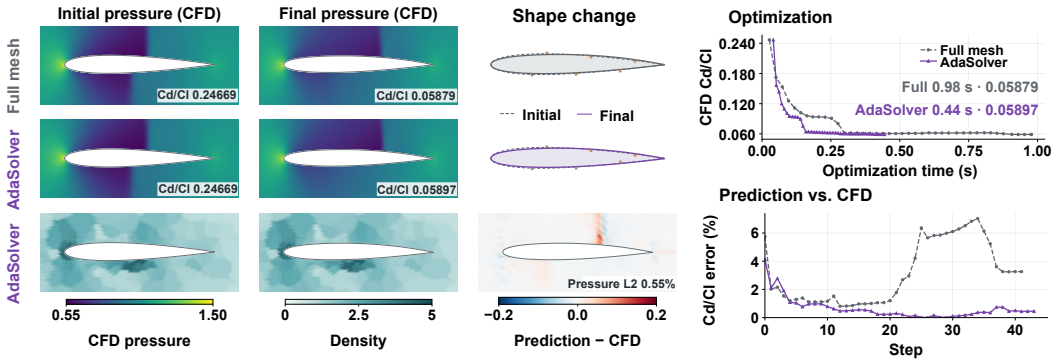

Figure 3: **Two-dimensional airfoil optimization.** Left: initial/final pressure, shape changes, AdaSolver density (sampling weights; uniform $= 1$), retained points, and pressure error. Right: CFD C_D/C_L versus online optimization time; relative ratio prediction error over accepted steps.

3D SuperWing. Figure 4 compares CMA-ES (Hansen & Ostermeier, 2001) optimization on SuperWing (Yang et al., 2025), with both methods initialized from the same geometry and flow condition. AdaSolver evaluates only 37.5% of the surface points and periodically refreshes the sampling density as the geometry evolves. Since nearby optimization iterates typically exhibit similar geometries, the same density can be reused across multiple steps, avoiding unnecessary resampling overhead. AdaSolver achieves a $1.86\times$ speedup while attaining a slightly lower CFD-validated C_D/C_L than full-mesh optimization. As shown in Figure 4, the relative prediction error also decreases from 3.82% to 3.30%.

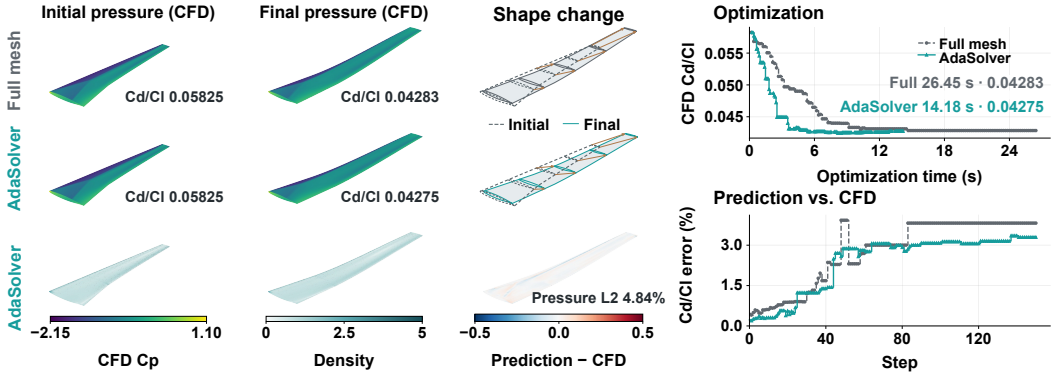

Figure 4: **SuperWing optimization.** Left: CFD pressure coefficients, shape changes, AdaSolver density (normalized as in Figure 3), retained points, and pressure error. Right: independently CFD-evaluated C_D/C_L versus optimization time; relative ratio prediction error over 150 generations.

5 Conclusion

AdaSolver turns spatial sampling density from a fixed discretization choice into a controllable design variable at test time. By compensating for the induced measure shift, it decouples where computation is allocated from the reference operator being approximated, establishing spatial sampling as a new axis of test-time scaling for point-based neural operators. Our experiments further show that effective sampling strategies can arise directly from computational-physics knowledge: classical refinement heuristics, geometric and flow features, and solution-dependent structures can all be translated into improved accuracy–compute trade-offs, and ultimately into faster surrogate-based aerodynamic design. More broadly, this perspective suggests a path toward measure-aware adaptive computation across a wider class of transformer-based neural operators, as well as training-time augmentation over diverse sampling distributions, allowing neural operators to scale not only in model capacity or iterative computation, but also in where computation is spent in physical space.

AI USE STATEMENT

Generative AI tools were used to assist with literature discovery and to edit and polish the manuscript. The authors checked the cited literature against the original sources and reviewed and revised the AI-assisted edits. All final research and editorial decisions were made by the authors, who take full responsibility for the content, accuracy, originality, and integrity of this work.

REPRODUCIBILITY STATEMENT

The theoretical derivations are provided in Appendix A. Dataset, training, evaluation, and inverse-design settings are detailed in Appendix B, while the complete task-specific sampling policies are provided in Appendix C. Additional ablations, computational costs, and qualitative results are reported in Appendix D.

REFERENCES

Benedikt Alkin, Andreas Furst, Simon Schmid, Lukas Gruber, Markus Holzleitner, and Johannes Brandstetter. Universal physics transformers: A framework for efficiently scaling neural operators. In *Advances in Neural Information Processing Systems*, volume 37, 2024.

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. *arXiv preprint arXiv:2408.11969*, 2024. doi: 10.48550/arXiv.2408.11969. URL https://arxiv.org/abs/2408.11969.

Julius Berner, Miguel Liu-Schiaffini, Jean Kossaifi, Valentin Duruisseaux, Boris Bonev, Kamyar Azizzadenesheli, and Anima Anandkumar. Principled approaches for extending neural architectures to function spaces for operator learning. *Nature Machine Intelligence*, 8:1173–1181, 2026.

Kevin Doherty, Colin Simpson, Stephen Becker, and Alireza Doostan. Quadconv: Quadrature-based convolutions with applications to non-uniform pde data compression. *Journal of Computational Physics*, 498:112636, 2024.

Wenbo Gao, Rui Xu, Yuxin Deng, and Yang Liu. Discretization-invariance? on the discretization mismatch errors in neural operators. In *International Conference on Learning Representations*, 2025.

Nikolaus Hansen and Andreas Ostermeier. Completely derandomized self-adaptation in evolution strategies. *Evolutionary Computation*, 9(2):159–195, 2001. doi: 10.1162/106365601750190398. URL https://doi.org/10.1162/106365601750190398.

Zhongkai Hao, Zhengyi Wang, Hang Su, Chengyang Ying, Yinpeng Dong, Songming Liu, Ze Cheng, Jian Song, and Jun Zhu. Gnot: A general neural operator transformer for operator learning. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pp. 12556–12569. PMLR, 2023. URL https://proceedings.mlr.press/v202/hao23c.html.

Pedro Hermosilla, Tobias Ritschel, Pere-Pau Vázquez, Àlvar Vinacua, and Timo Ropinski. Monte carlo convolution for learning on non-uniformly sampled point clouds. *ACM Transactions on Graphics*, 37(6), 2018.

Kai Huang, Martin Krugener, Adam Brown, Felix Menhorn, Hans-Joachim Bungartz, and Dirk Hartmann. Machine learning-based optimal mesh generation in computational fluid dynamics. *arXiv preprint arXiv:2102.12923*, 2021.

Ben Jenkins. Adaptive test-time compute allocation for neural pde solvers. In *International Conference on Learning Representations*, 2026.

Byeong Joo Kim, Masaki Kawano, Yusuke Iwasawa, and Yutaka Matsuo. Quadnorm: Resolution-robust normalization for neural operators. *arXiv preprint arXiv:2605.07375*, 2026.

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

Zongyi Li, Daniel Zhengyu Huang, Burigede Liu, and Anima Anandkumar. Fourier neural operator with learned deformations for pdes on general geometries. *Journal of Machine Learning Research*, 24(388):1–26, 2023a. URL https://jmlr.org/papers/v24/23-0064.html.

Zongyi Li, Nikola Kovachki, Christopher Choy, Boyi Li, Jean Kossaifi, Shourya Otta, Mohammad Amin Nabian, Markus Stadler, Christian Hundt, Kamyar Azizzadenesheli, and Anima Anandkumar. Geometry-informed neural operator for large-scale 3d pdes. In *Advances in Neural Information Processing Systems*, volume 36, 2023b.

Phillip Lippe, Bastiaan S. Veeling, Paris Perdikaris, Richard E. Turner, and Johannes Brandstetter. PDE-refiner: Achieving accurate long rollouts with neural PDE solvers. In *Advances in Neural Information Processing Systems*, volume 36, 2023.

Sifan Liu, Zhongkai Hao, Chengyang Ying, Hang Su, Zhijie Cheng, and Jun Zhu. Nuno: A general framework for learning parametric pdes with non-uniform data. In *Proceedings of the 40th International Conference on Machine Learning*, 2023.

Huakun Luo, Haixu Wu, Hang Zhou, Lei Xing, Yide Di, Jianmin Wang, and Mingsheng Long. Transolver++: An accurate neural solver for pdes on million-scale geometries. In *Proceedings of the 42nd International Conference on Machine Learning*, 2025.

Oriol Obiols-Sales, Abhinav Vishnu, Nicholas Malaya, and Aparna Chandramowlishwaran. Adarnet: Deep learning predicts adaptive mesh refinement. In *International Conference on Parallel Processing*, pp. 524–534, 2023.

Jack Rowbottom, Georg Maierhofer, Thomas Deveney, Eike Mueller, Alberto Paganini, Katharina Schratz, Pietro Lio, Carola-Bibiane Schonlieb, and Chris Budd. G-adaptivity: Optimised graph-based mesh relocation for finite element methods. *arXiv preprint arXiv:2407.04516*, 2024.

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

Charlie Snell, Jaehoon Lee, Kelvin Xu, and Aviral Kumar. Scaling llm test-time compute optimally can be more effective than scaling model parameters. *arXiv preprint arXiv:2408.03314*, 2024. URL https://arxiv.org/abs/2408.03314.

Nobuyuki Umetani and Bernd Bickel. Learning three-dimensional flow for interactive aerodynamic design. *ACM Transactions on Graphics*, 37(4), 2018. doi: 10.1145/3197517.3201325. URL https://doi.org/10.1145/3197517.3201325.

Clinton J. Wang and Polina Golland. Discretization invariant networks for learning maps between neural fields. *arXiv preprint arXiv:2206.01178*, 2022.

Hanwen Wang and Paris Perdikaris. Adaptive tokenization for vision transformer pde simulation. In *AI and PDE Workshop at the International Conference on Learning Representations*, 2026.

Haixu Wu, Huakun Luo, Haowen Wang, Jianmin Wang, and Mingsheng Long. Transolver: A fast transformer solver for PDEs on general geometries. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*. PMLR, 2024.

Yunjia Yang, Weishao Tang, Mengxin Liu, Nils Thuerey, Yufei Zhang, and Haixin Chen. SuperWing: A comprehensive transonic wing dataset for data-driven aerodynamic design. *arXiv preprint arXiv:2512.14397*, 2025. doi: 10.48550/arXiv.2512.14397. URL https://arxiv.org/abs/2512.14397.

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. In *Proceedings of the 43rd International Conference on Machine Learning*, volume 306 of *Proceedings of Machine Learning Research*. PMLR, 2026. URL https://arxiv.org/abs/2602.04940.

A SAMPLING-MEASURE BIAS IN PHYSICS-ATTENTION

We analyze the effect of evaluating a frozen Physics-Attention layer under a sampling measure different from the one used during training. Let (Ω, ρ) be the physical domain equipped with the reference probability measure, let $h : \Omega \to \mathbb{R}^C$ be the point-feature field, and let Ω_s be a continuous slice domain of finite measure. A bounded measurable slice-weight function $w : \Omega \times \Omega_s \to [0, \infty)$ satisfies

$$\int_{\Omega_s} w(x, \xi_s)\, \mathrm{d}\xi_s = 1, \qquad \int_{\Omega} w(x, \xi_s)\, \mathrm{d}\rho(x) > 0. \tag{A.1}$$

For each $\xi_s \in \Omega_s$, define the slice-weighted probability measure

$$\mathrm{d}\pi_{\xi_s}(x) := \frac{w(x, \xi_s)}{\int_\Omega w(y, \xi_s)\, \mathrm{d}\rho(y)}\, \mathrm{d}\rho(x). \tag{A.2}$$

Let $\widetilde{\rho}$ be the inference-time sampling measure. We assume $\widetilde{\rho} \ll \rho$ and write

$$\lambda := \frac{\mathrm{d}\widetilde{\rho}}{\mathrm{d}\rho} > 0 \quad \rho\text{-a.e.} \tag{A.3}$$

The reference and shifted physics-token fields are

$$u_s^{\rho}(\xi_s) := \mathbb{E}_{\pi_{\xi_s}}[h], \qquad u_s^{\widetilde{\rho}}(\xi_s) := \frac{\mathbb{E}_{\pi_{\xi_s}}[\lambda h]}{\mathbb{E}_{\pi_{\xi_s}}[\lambda]}. \tag{A.4}$$

Since $\lambda > 0$ holds ρ-almost everywhere and w is bounded with positive ρ-mass,

$$\mathbb{E}_{\pi_{\xi_s}}[\lambda] = \frac{\int_\Omega w(x, \xi_s)\, \mathrm{d}\widetilde{\rho}(x)}{\int_\Omega w(x, \xi_s)\, \mathrm{d}\rho(x)} \in (0, \infty).$$

Hence, $u_s^{\widetilde{\rho}}$ is well defined, and its discrepancy from u_s^{ρ} reflects a change in the limiting tokenization operator rather than finite-sample quadrature error.

Proposition A.1 (Exact bias of a physics token)**.** *Fix $\xi_s \in \Omega_s$. Assume $h \in L^2(\pi_{\xi_s}; \mathbb{R}^C)$ and $\lambda h \in L^1(\pi_{\xi_s}; \mathbb{R}^C)$. Then*

$$u_s^{\widetilde{\rho}}(\xi_s) - u_s^{\rho}(\xi_s) = \frac{\mathrm{Cov}_{\pi_{\xi_s}}(\lambda, h)}{\mathbb{E}_{\pi_{\xi_s}}[\lambda]} = \mathbb{E}_{\pi_{\xi_s}}\left[\left(\frac{\lambda}{\mathbb{E}_{\pi_{\xi_s}}[\lambda]} - 1\right) h\right], \tag{A.5}$$

where

$$\mathrm{Cov}_{\pi_{\xi_s}}(\lambda, h) := \mathbb{E}_{\pi_{\xi_s}}\left[(\lambda - \mathbb{E}_{\pi_{\xi_s}}[\lambda])\left(h - \mathbb{E}_{\pi_{\xi_s}}[h]\right)\right] \in \mathbb{R}^C. \tag{A.6}$$

Furthermore, if $\lambda \in L^2(\pi_{\xi_s})$, define the bounded linear operators from $L^2(\pi_{\xi_s}; \mathbb{R}^C)$ to $\mathbb{R}^C$ by

$$\mathcal{P}_{\xi_s}^{\rho} f := \mathbb{E}_{\pi_{\xi_s}}[f], \qquad \mathcal{P}_{\xi_s}^{\widetilde{\rho}} f := \frac{\mathbb{E}_{\pi_{\xi_s}}[\lambda f]}{\mathbb{E}_{\pi_{\xi_s}}[\lambda]}. \tag{A.7}$$

Their operator-norm discrepancy is exactly

$$\left\| \mathcal{P}_{\xi_s}^{\widetilde{\rho}} - \mathcal{P}_{\xi_s}^{\rho} \right\|_{\mathrm{op}} = \left\| \frac{\lambda}{\mathbb{E}_{\pi_{\xi_s}}[\lambda]} - 1 \right\|_{L^2(\pi_{\xi_s})} = \frac{\sqrt{\mathrm{Var}_{\pi_{\xi_s}}(\lambda)}}{\mathbb{E}_{\pi_{\xi_s}}[\lambda]}. \tag{A.8}$$

Proof. By equation A.4,

$$u_s^{\widetilde{\rho}}(\xi_s) - u_s^{\rho}(\xi_s) = \frac{\mathbb{E}_{\pi_{\xi_s}}[\lambda h] - \mathbb{E}_{\pi_{\xi_s}}[\lambda]\mathbb{E}_{\pi_{\xi_s}}[h]}{\mathbb{E}_{\pi_{\xi_s}}[\lambda]} = \frac{\mathrm{Cov}_{\pi_{\xi_s}}(\lambda, h)}{\mathbb{E}_{\pi_{\xi_s}}[\lambda]}, \tag{A.9}$$

which also equals the second expression in equation A.5.

For the operator statement, set

$$r_{\xi_s} := \frac{\lambda}{\mathbb{E}_{\pi_{\xi_s}}[\lambda]} - 1. \tag{A.10}$$

Then, for every $f \in L^2(\pi_{\xi_s}; \mathbb{R}^C)$,

$$\left(\mathcal{P}^{\widetilde{\rho}}_{\xi_s} - \mathcal{P}^{\rho}_{\xi_s}\right) f = \mathbb{E}_{\pi_{\xi_s}}[r_{\xi_s} f]. \tag{A.11}$$

Cauchy–Schwarz gives

$$\left\|\mathbb{E}_{\pi_{\xi_s}}[r_{\xi_s} f]\right\|_2 \le \|r_{\xi_s}\|_{L^2(\pi_{\xi_s})} \|f\|_{L^2(\pi_{\xi_s};\mathbb{R}^C)}, \tag{A.12}$$

so the operator norm is at most $\|r_{\xi_s}\|_{L^2(\pi_{\xi_s})}$. If $r_{\xi_s} \neq 0$, choose a unit vector $e \in \mathbb{R}^C$ and

$$f^\star(x) := \frac{r_{\xi_s}(x)}{\|r_{\xi_s}\|_{L^2(\pi_{\xi_s})}} e. \tag{A.13}$$

Then $\|f^\star\|_{L^2(\pi_{\xi_s};\mathbb{R}^C)} = 1$ and

$$\left(\mathcal{P}^{\widetilde{\rho}}_{\xi_s} - \mathcal{P}^{\rho}_{\xi_s}\right) f^\star = \|r_{\xi_s}\|_{L^2(\pi_{\xi_s})} e, \tag{A.14}$$

which attains the upper bound. The case $r_{\xi_s} = 0$ is immediate. Finally,

$$\|r_{\xi_s}\|^2_{L^2(\pi_{\xi_s})} = \frac{\mathbb{E}_{\pi_{\xi_s}}\left[(\lambda - \mathbb{E}_{\pi_{\xi_s}}[\lambda])^2\right]}{\mathbb{E}_{\pi_{\xi_s}}[\lambda]^2} = \frac{\mathrm{Var}_{\pi_{\xi_s}}(\lambda)}{\mathbb{E}_{\pi_{\xi_s}}[\lambda]^2}, \tag{A.15}$$

which proves equation A.8. □

A.1 Propagation through Continuous Physics-Attention

We first recall the continuous formulation leading to equation 13 of Transolver (Wu et al., 2024). Let $g : \Omega \to \Omega_s$ be a diffeomorphism from the physical domain to the slice domain, with

$$J_g(\xi_s) := \left|\det\left(\nabla_{\xi_s} g^{-1}(\xi_s)\right)\right|. \tag{A.16}$$

For a bounded physics-token field $u_s : \Omega_s \to \mathbb{R}^C$, the slice-to-slice Physics-Attention kernel is

$$\begin{aligned}
\ell_{u_s}(\xi'_s, \xi_s) &:= \frac{(W_q u_s(\xi'_s))^\top W_k u_s(\xi_s)}{\tau}, \\
\alpha_{u_s}(\xi'_s, \xi_s) &:= \frac{\exp(\ell_{u_s}(\xi'_s, \xi_s))}{\int_{\Omega_s} \exp(\ell_{u_s}(\xi'_s, \zeta_s))\, \mathrm{d}\zeta_s}, \\
\kappa_{ss,u_s}(\xi'_s, \xi_s) &:= \alpha_{u_s}(\xi'_s, \xi_s) W_v,
\end{aligned} \tag{A.17}$$

where $W_q, W_k \in \mathbb{R}^{d\times C}$, $W_v \in \mathbb{R}^{C\times C}$, and $\tau > 0$ are fixed. Thus, κ_{ss,u_s} is the matrix-valued kernel that performs attention among physics tokens. After expressing the physical-to-slice kernel as a weighted combination of κ_{ss,u_s}, the continuous Physics-Attention operator becomes

$$\begin{aligned}
\mathcal{T}[u_s](x) &:= \int_{\Omega_s} w(x, \xi'_s) \int_{\Omega_s} \kappa_{ss,u_s}(\xi'_s, \xi_s) u_s(\xi_s) J_g(\xi_s)\, \mathrm{d}\xi_s\, \mathrm{d}\xi'_s \\
&= \int_{\Omega_s} w(x, \xi'_s) \int_{\Omega_s} \alpha_{u_s}(\xi'_s, \xi_s) W_v u_s(\xi_s) J_g(\xi_s)\, \mathrm{d}\xi_s\, \mathrm{d}\xi'_s.
\end{aligned} \tag{A.18}$$

This is the deslice–attention–token factorization in the continuous derivation of Transolver. If the slice domain is represented by M tokens $s_1, \ldots, s_M$, with $q_j = W_q s_j$, $k_j = W_k s_j$, and $v_j = W_v s_j$, its Monte Carlo discretization is

$$h'_i = \sum_{j=1}^{M} w_{i,j} \sum_{t=1}^{M} \frac{\exp(q_j^\top k_t/\tau)}{\sum_{p=1}^{M} \exp(q_j^\top k_p/\tau)} v_t, \qquad i = 1, \ldots, N, \tag{A.19}$$

which is equation 13 of Transolver under the simplifying choice $J_g \equiv 1$. For the token fields in equation A.4, we write $\mathcal{G}_\rho(h) := \mathcal{T}[u^\rho_s]$ and $\mathcal{G}_{\widetilde{\rho}}(h) := \mathcal{T}[u^{\widetilde{\rho}}_s]$. The map g, the slice weight w, and the projection matrices are held fixed when the sampling measure changes.

Lemma A.2 (Exact output perturbation for a token shift)**.** *Let $u_s, \delta_s \in L^\infty(\Omega_s; \mathbb{R}^C)$. The logit perturbation induced by replacing u_s with $u_s + \delta_s$ is*

$$\begin{aligned}\Delta\ell_{\delta_s}(\xi_s', \xi_s) &:= \ell_{u_s+\delta_s}(\xi_s', \xi_s) - \ell_{u_s}(\xi_s', \xi_s) \\ &= \frac{(W_q\delta_s(\xi_s'))^\top W_k u_s(\xi_s) + (W_q u_s(\xi_s'))^\top W_k \delta_s(\xi_s) + (W_q\delta_s(\xi_s'))^\top W_k \delta_s(\xi_s)}{\tau}.\end{aligned} \tag{A.20}$$

For each query ξ_s', define

$$R_{\delta_s}(\xi_s', \xi_s) := \frac{\exp(\Delta\ell_{\delta_s}(\xi_s', \xi_s))}{\int_{\Omega_s} \alpha_{u_s}(\xi_s', \zeta_s)\exp(\Delta\ell_{\delta_s}(\xi_s', \zeta_s))\,\mathrm{d}\zeta_s}. \tag{A.21}$$

Then $\alpha_{u_s+\delta_s} = \alpha_{u_s} R_{\delta_s}$, and

$$\begin{aligned}\mathcal{T}[u_s + \delta_s](x) - \mathcal{T}[u_s](x) = \int_{\Omega_s} w(x, \xi_s') \int_{\Omega_s} \alpha_{u_s}(\xi_s', \xi_s)\Big[&(R_{\delta_s}(\xi_s', \xi_s) - 1)W_v u_s(\xi_s) \\ &+ R_{\delta_s}(\xi_s', \xi_s) W_v \delta_s(\xi_s)\Big] J_g(\xi_s)\,\mathrm{d}\xi_s\,\mathrm{d}\xi_s'.\end{aligned} \tag{A.22}$$

The two terms are the attention-redistribution bias and the value-token bias, respectively.

Proof. By equation A.20,

$$\begin{aligned}\alpha_{u_s+\delta_s}(\xi_s', \xi_s) &= \frac{\exp(\ell_{u_s}(\xi_s', \xi_s))\exp(\Delta\ell_{\delta_s}(\xi_s', \xi_s))}{\int_{\Omega_s}\exp(\ell_{u_s}(\xi_s', \zeta_s))\exp(\Delta\ell_{\delta_s}(\xi_s', \zeta_s))\,\mathrm{d}\zeta_s} \\ &= \alpha_{u_s}(\xi_s', \xi_s) R_{\delta_s}(\xi_s', \xi_s).\end{aligned} \tag{A.23}$$

Substituting this identity and $u_s + \delta_s$ into equation A.18, then subtracting $\mathcal{T}[u_s]$, gives equation A.22. □

Theorem A.3 (Exact propagation of sampling-measure bias)**.** *Assume the conditions of Proposition A.1 hold for every $\xi_s \in \Omega_s$, and define the token perturbation*

$$\delta_s^{\widetilde{\rho}}(\xi_s) := u_s^{\widetilde{\rho}}(\xi_s) - u_s^{\rho}(\xi_s) = \frac{\mathrm{Cov}_{\pi_{\xi_s}}(\lambda, h)}{\mathbb{E}_{\pi_{\xi_s}}[\lambda]}. \tag{A.24}$$

If $u_s^\rho, \delta_s^{\widetilde{\rho}} \in L^\infty(\Omega_s; \mathbb{R}^C)$, then

$$\begin{aligned}\mathcal{G}_{\widetilde{\rho}}(h)(x) - \mathcal{G}_\rho(h)(x) = \int_{\Omega_s} w(x, \xi_s') \int_{\Omega_s} \alpha_{u_s^\rho}(\xi_s', \xi_s)\Big[&(R_{\delta_s^{\widetilde{\rho}}}(\xi_s', \xi_s) - 1)W_v u_s^\rho(\xi_s) \\ &+ R_{\delta_s^{\widetilde{\rho}}}(\xi_s', \xi_s) W_v \frac{\mathrm{Cov}_{\pi_{\xi_s}}(\lambda, h)}{\mathbb{E}_{\pi_{\xi_s}}[\lambda]}\Big] J_g(\xi_s)\,\mathrm{d}\xi_s\,\mathrm{d}\xi_s',\end{aligned} \tag{A.25}$$

where $R_{\delta_s^{\widetilde{\rho}}}$ is given by equation A.21 with base field $u_s = u_s^\rho$. Thus, the exact output bias is the sum of the change in attention weights and the direct propagation of the biased value tokens.

Proof. Equation A.24 follows from Proposition A.1. Applying Lemma A.2 with $u_s = u_s^\rho$ and $\delta_s = \delta_s^{\widetilde{\rho}}$ gives equation A.25. □

We next consider a small change of sampling measure,

$$\mathrm{d}\rho_\varepsilon(x) = (1 + \varepsilon\eta(x))\,\mathrm{d}\rho(x), \qquad \mathbb{E}_\rho[\eta] = 0, \tag{A.26}$$

where $\eta \in L^\infty(\rho)$ and $|\varepsilon|$ is small enough that $1 + \varepsilon\eta > 0$. Let $u_s^{\rho_\varepsilon}$ be the token field under ρ_ε. Proposition A.1 gives

$$u_s^{\rho_\varepsilon}(\xi_s) - u_s^\rho(\xi_s) = \frac{\varepsilon\,\mathrm{Cov}_{\pi_{\xi_s}}(\eta, h)}{1 + \varepsilon\mathbb{E}_{\pi_{\xi_s}}[\eta]} = \varepsilon\,\mathrm{Cov}_{\pi_{\xi_s}}(\eta, h) + \mathcal{O}(\varepsilon^2), \tag{A.27}$$

uniformly in ξ_s. The corresponding first-order change of the attention logit is

$$\begin{aligned}\ell_\eta^{(1)}(\xi_s', \xi_s) := \frac{1}{\tau}\Big[&\big(W_q\,\mathrm{Cov}_{\pi_{\xi_s'}}(\eta, h)\big)^\top W_k u_s^\rho(\xi_s) \\ &+ \big(W_q u_s^\rho(\xi_s')\big)^\top W_k\,\mathrm{Cov}_{\pi_{\xi_s}}(\eta, h)\Big].\end{aligned} \tag{A.28}$$

B Implementation Details

We describe the governing equations, data splits, and forward operators for the benchmarks used in our experiments. All sampling budgets are measured relative to the reference point sets specified below.

B.1 Standard Benchmarks

Elasticity and Plasticity follow the solid-mechanics benchmarks in Geo-FNO (Li et al., 2023a). Both satisfy momentum balance,

$$\rho_s \frac{\partial^2 \mathbf{u}}{\partial t^2} = \nabla \cdot \boldsymbol{\sigma}, \tag{B.1}$$

where ρ_s is material density, $\mathbf{u}$ is displacement, and $\boldsymbol{\sigma}$ is stress. The two tasks use different constitutive laws to describe the material response.

Elasticity. This task predicts the equilibrium stress in a unit square containing an irregular central hole. The bottom edge is fixed, and the top edge is pulled upward. The material follows the incompressible Rivlin–Saunders model, with strain energy

$$W = C_1(I_1 - 3) + C_2(I_2 - 3), \tag{B.2}$$

where I_1 and I_2 are deformation invariants, $C_1 = 1.863 \times 10^5$, and $C_2 = 9.79 \times 10^3$. The hole radius varies between 0.2 and 0.4, and each geometry contains 972 unstructured points. The operator maps the geometry to the von Mises stress field, $\mathcal{G}_{\text{elas}} : \Omega \mapsto \sigma_{\text{vM}}(\cdot)$. We use 1,000 training, 800 validation, and 200 test geometries.

Plasticity. This task models a metal block compressed by a rigid, frictionless die with profile S_d. It follows equation B.1 with an elastoplastic stress law,

$$\boldsymbol{\sigma} = \mathbb{C} : (\boldsymbol{\varepsilon} - \boldsymbol{\varepsilon}^p), \tag{B.3}$$

where $\mathbb{C}$ is the elastic stiffness tensor, and $\boldsymbol{\varepsilon}$ and $\boldsymbol{\varepsilon}^p$ are total and plastic strain. The material obeys the von Mises yield criterion, with Young's modulus $200\,\text{GPa}$, Poisson's ratio 0.3, yield strength $70\,\text{MPa}$, and density $7850\,\text{kg}\,\text{m}^{-3}$. The $50\,\text{mm} \times 15\,\text{mm}$ block is fixed at the bottom and compressed at $3\,\text{m}\,\text{s}^{-1}$. We use 800 training, 107 validation, and 80 test cases. Each case contains a 101×31 mesh and 20 time steps. The operator maps the die profile to the displacement sequence, $\mathcal{G}_{\text{plas}} : S_d \mapsto \{\mathbf{u}(\cdot, t_j)\}_{j=1}^{20}$, predicting both displacement components at all time steps in one pass.

Airfoil. The Airfoil benchmark (Li et al., 2023a) describes transonic flow around deformations of a NACA-0012 airfoil. The steady inviscid flow satisfies the compressible Euler equations,

$$\begin{aligned} \nabla \cdot (\rho_f \mathbf{v}) = 0, \quad \nabla \cdot (\rho_f \mathbf{v} \otimes \mathbf{v} + p\mathbf{I}) = 0, \\ \nabla \cdot [(\mathcal{E} + p)\mathbf{v}] = 0, \qquad\qquad p = (\gamma - 1)\left(\mathcal{E} - \tfrac{1}{2}\rho_f \|\mathbf{v}\|^2\right), \end{aligned} \tag{B.4}$$

where ρ_f, $\mathbf{v}$, p, and $\mathcal{E}$ denote fluid density, velocity, pressure, and total energy per unit volume. The freestream conditions are $\rho_\infty = p_\infty = 1$, $M_\infty = 0.8$, and zero angle of attack, with $\gamma = 1.4$ and no penetration at the airfoil surface. Each case uses a 221×51 body-fitted mesh with 11,271 points. The dataset split contains 1,000 training and 200 test cases. Our operator maps mesh coordinates to pressure, $\mathcal{G}_{\text{airfoil}} : \Omega \mapsto p(\cdot)$, from which surface integration provides C_D/C_L. Both reported metrics use the 110 test cases with reference pressure-force $C_L > 0.02$. This threshold was tightened from 0.01 in a post-hoc sensitivity analysis, excluding two of the original 112 positive-lift cases.

Pipe. The Pipe benchmark (Li et al., 2023a) considers incompressible viscous flow through two-dimensional pipes with varying centerlines. The governing Navier–Stokes equations are

$$\frac{\partial \mathbf{v}}{\partial t} + (\mathbf{v} \cdot \nabla)\mathbf{v} = -\nabla p + \nu \Delta \mathbf{v}, \qquad \nabla \cdot \mathbf{v} = 0, \tag{B.5}$$

where p is pressure divided by the constant density and $\nu = 0.005$. The inlet has a parabolic velocity profile with unit peak speed; the walls satisfy no-slip conditions, and the outlet has a free boundary condition. Pipes have length 10 and width 1, with centerlines parameterized by piecewise cubic polynomials. The dataset contains 1,000 training and 200 test geometries, each discretized on a 129×129 mesh with 16,641 points. The forward operator $\mathcal{G}_{\text{pipe}} : \Omega \mapsto v_x(\cdot)$ maps the two-dimensional mesh coordinates to the steady horizontal velocity field.

Darcy. The Darcy benchmark (Li et al., 2021) models steady flow through a heterogeneous porous medium on $\Omega = (0,1)^2$,

$$-\nabla\cdot\big(a(x)\nabla p(x)\big) = 1 \quad \text{in } \Omega, \qquad p = 0 \quad \text{on } \partial\Omega, \tag{B.6}$$

where a is the permeability coefficient and p is pressure. The permeability field is generated by thresholding a Gaussian random field $z \sim \mathcal{N}(0, (-\Delta + 9I)^{-2})$, with $a(x) = 12$ for $z(x) \geq 0$ and $a(x) = 3$ otherwise. Solutions are generated on a 421×421 grid and subsampled to an 85×85 reference grid of 7,225 points. We use 1,000 training and 200 test cases. The operator $\mathcal{G}_{\mathrm{Darcy}} : a(\cdot) \mapsto p(\cdot)$ maps the permeability field to pressure.

B.2 Industrial Applications

The industrial tasks predict aerodynamic fields on three-dimensional vehicle and wing geometries. Drag and lift are obtained by integrating the predicted surface forces.

ShapeNet-Car. ShapeNet-Car (Umetani & Bickel, 2018; Wu et al., 2024) contains 889 car geometries simulated at a driving speed of $72\,\mathrm{km\,h^{-1}}$. The underlying three-dimensional flow is governed by the incompressible Navier–Stokes equations,

$$\nabla\cdot\mathbf{v} = 0, \qquad \rho_f\left(\frac{\partial\mathbf{v}}{\partial t} + (\mathbf{v}\cdot\nabla)\mathbf{v}\right) = -\nabla p + \mu\Delta\mathbf{v}, \tag{B.7}$$

where μ is dynamic viscosity. Each case contains 32,186 points covering the vehicle surface and surrounding volume. We use 710 training, 79 validation, and 100 test geometries. The operator maps coordinates, signed distance, and surface normals to velocity and pressure, $\mathcal{G}_{\mathrm{car}} : (\mathbf{x}, \mathrm{SDF}, \mathbf{n}) \mapsto (\mathbf{v}, p)$. Velocity is predicted over the full point cloud and pressure on the surface; both fields contribute to drag estimation.

DrivAerML. DrivAerML (Ashton et al., 2024) provides high-fidelity automotive aerodynamics for 500 deformations of the DrivAer vehicle. Its hybrid Reynolds-averaged Navier–Stokes/large-eddy simulations (RANS–LES) solve incompressible flow with modeled turbulent stresses. The resolved equations can be written as

$$\nabla\cdot\overline{\mathbf{v}} = 0, \qquad \rho_f\left(\frac{\partial\overline{\mathbf{v}}}{\partial t} + (\overline{\mathbf{v}}\cdot\nabla)\overline{\mathbf{v}}\right) = -\nabla\overline{p} + \nabla\cdot\left(2\mu\overline{\mathbf{S}} - \rho_f\boldsymbol{\tau}_t\right), \tag{B.8}$$

where $\overline{\mathbf{S}}$ is the resolved strain-rate tensor and $\boldsymbol{\tau}_t$ represents unresolved turbulent stresses. We use the surface data at $U_\infty = 38.889\,\mathrm{m\,s^{-1}}$, with about 8.8 million points per test geometry. Following the available split used with Transolver-3 (Zhou et al., 2026), we use 400 training, 34 validation, and 50 test cases. Separate operators map coordinates and surface normals to pressure and wall shear stress, $\mathcal{G}_{\mathrm{DrivAer}} : (\mathbf{x}, \mathbf{n}) \mapsto (p, \boldsymbol{\tau}_w)$, which are integrated to estimate C_D.

SuperWing. SuperWing (Yang et al., 2025) contains 28,856 flow conditions over 4,239 transonic swept-wing geometries with variations in planform, twist, dihedral, and airfoil shape. The flow is governed by the steady compressible RANS equations,

$$\begin{aligned} \nabla\cdot(\rho_f\mathbf{v}) &= 0, \\ \nabla\cdot(\rho_f\mathbf{v}\otimes\mathbf{v} + p\mathbf{I}) &= \nabla\cdot\boldsymbol{\tau}_{\mathrm{eff}}, \\ \nabla\cdot[(\mathcal{E}+p)\mathbf{v}] &= \nabla\cdot(\boldsymbol{\tau}_{\mathrm{eff}}\mathbf{v} - \mathbf{q}_{\mathrm{eff}}), \end{aligned} \tag{B.9}$$

where the variables denote mean-flow quantities, and $\boldsymbol{\tau}_{\mathrm{eff}}$ and $\mathbf{q}_{\mathrm{eff}}$ include molecular and modeled turbulent transport. ADflow generates the solutions at Mach numbers in $[0.75, 0.90]$ and angles of attack in $[2^\circ, 12^\circ]$, with Reynolds number 2×10^7 and freestream temperature $300\,\mathrm{K}$.

Each wing is represented by $128 \times 256 = 32{,}768$ surface points. We use 23,366 training, 2,619 validation, and 2,871 test cases, with no wing geometry shared across splits. The operator maps wing geometry, Mach number, and angle of attack to surface pressure and skin-friction coefficients, $\mathcal{G}_{\mathrm{wing}} : (\Gamma, M_\infty, \alpha) \mapsto (C_p, C_{f,\tau}, C_{f,z})$, where τ and z denote the streamwise and spanwise friction directions. These predicted fields determine the drag-to-lift ratio C_D/C_L.

B.3 METRICS

Drag and lift coefficients. Given the aerodynamic force $\mathbf{F}$, drag and lift coefficients are

$$C_D = \frac{\mathbf{F}\cdot\mathbf{e}_D}{q_\infty A_{\text{ref}}}, \qquad C_L = \frac{\mathbf{F}\cdot\mathbf{e}_L}{q_\infty A_{\text{ref}}}, \qquad q_\infty = \tfrac{1}{2}\rho_\infty U_\infty^2, \tag{B.10}$$

where $\mathbf{e}_D$ and $\mathbf{e}_L$ are the drag and lift directions, and A_{ref} is the reference area. We reconstruct sampled predictions on the full reference surface before force integration. For density design based on the drag-to-lift ratio C_D/C_L, we exclude cases with near-zero reference lift to avoid unstable ratios. This is consistent with practical aerodynamic design, where optimization is typically initialized from a feasible lifting configuration.

DrivAerML and ShapeNet-Car. Both car benchmarks include pressure and viscous contributions to drag. DrivAerML predicts pressure p and wall shear stress $\boldsymbol{\tau}_w$ directly, while ShapeNet-Car estimates the viscous contribution from predicted velocity. Following their respective normal and traction conventions, the implemented coefficients are

$$\begin{aligned} C_D^{\text{DrivAerML}} &= \frac{\sum_i A_i\big(p_i n_{i,x} - \tau_{w,i,x}\big)}{q_\infty A_{\text{ref}}}, \\ C_D^{\text{ShapeNet-Car}} &= -\frac{\sum_i A_i n_{i,z}\big(\bar{p}_i + \mu g_{i,z}\big)}{q_\infty A_{\text{ref}}}. \end{aligned} \tag{B.11}$$

Here A_i and $\mathbf{n}_i$ are surface-cell areas and normals. For ShapeNet-Car, $\bar{p}_i$ averages the four vertex pressures, and $g_{i,z}$ is the discrete gradient estimate of the streamwise velocity from the benchmark's quadrilateral surface stencil. DrivAerML uses its supplied reference area, whereas ShapeNet-Car uses the area of the surface's convex-hull projection onto the x–y plane. The normalization uses $(\rho_\infty, U_\infty) = (1, 38.889)$ for DrivAerML and $(0.3, 20)$ for ShapeNet-Car, with $\mu = 1.8 \times 10^{-5}$ for the latter.

Airfoil. The inviscid Airfoil benchmark (Li et al., 2023a) uses pressure forces alone. For ordered surface nodes (x_i, y_i), define $\Delta x_i = x_{i+1} - x_i$, $\Delta y_i = y_{i+1} - y_i$, and $\bar{p}_i = (p_i + p_{i+1})/2$. Trapezoidal integration along the surface gives

$$C_D = -\frac{\sum_{i=1}^{n_s-1} \bar{p}_i \Delta y_i}{q_\infty c}, \qquad C_L = \frac{\sum_{i=1}^{n_s-1} \bar{p}_i \Delta x_i}{q_\infty c}, \tag{B.12}$$

where $n_s = 121$, the reference chord is $c = 1$, and $q_\infty = 0.448$. The drag and lift directions coincide with the x and y axes because the angle of attack is zero.

SuperWing. SuperWing (Yang et al., 2025) integrates pressure and skin-friction coefficients on the wing surface. We first undo the stored friction-channel scaling factors of 150 and 300. Using the dataset's surface-normal convention, the dimensionless force vector is

$$\mathbf{C}_F = \frac{1}{A_{\text{ref}}} \sum_i A_i \left[C_{p,i}\mathbf{n}_i + (\mathbf{I} - \mathbf{n}_i\mathbf{n}_i^\top)\big(C_{f,\tau,i}\mathbf{t}_i + C_{f,z,i}\mathbf{e}_z\big) \right], \tag{B.13}$$

where $\mathbf{t}_i$ is the local chordwise direction in the x–y plane, and A_{ref} is the half-wing reference area. The tangential projection converts the stored friction components into surface traction. Rotating by the angle of attack α gives

$$C_D = C_{F,x}\cos\alpha + C_{F,y}\sin\alpha, \qquad C_L = -C_{F,x}\sin\alpha + C_{F,y}\cos\alpha. \tag{B.14}$$

B.4 Training Details

All backbones follow the Transolver architecture (Wu et al., 2024), with eight Physics-Attention blocks and eight attention heads per block. Table 3 summarizes the task-specific model and training parameters. We use AdamW with weight decay 10^{-5} and gradient-norm clipping at 0.1. Experiments run on a server equipped with eight NVIDIA H20 GPUs and two Intel Xeon Platinum 8457C processors, providing 96 physical CPU cores and 192 hardware threads.

Table 3: Backbone configurations for the eight benchmarks. Batch sizes are global numbers of cases per optimization step. Learning rates are the initial rate for cosine decay and the peak rate for OneCycle. Sampling bounds are inclusive point counts.

	Standard PDE benchmarks					Industrial applications		
Parameter	Elasticity	Plasticity	Airfoil	Pipe	Darcy	ShapeNet-Car	DrivAerML	SuperWing
Hidden width	128	128	128	128	128	256	256	128
Physics states	64	64	64	64	64	32	32	64
MLP ratio	2	1	2	2	2	2	2	2
Learning rate	10^{-3}	10^{-3}	10^{-3}	10^{-3}	10^{-3}	10^{-3}	2×10^{-3}	10^{-3}
Epochs	500	1,000	1,000	1,000	1,000	400	1,000	1,000
Batch size	1	8	8	4	4	2	4	16
Scheduler	Cosine	OneCycle	OneCycle	OneCycle	OneCycle	OneCycle	OneCycle	OneCycle
Loss	Rel. L_2	Rel. L_2	Rel. L_2	Rel. L_2	Rel. L_2	Dual MSE	Rel. L_2	Rel. L_2
Min. points	389	1,252	4,508	6,656	7,225	12,874	85,900	9,830
Max. points	972	3,131	11,271	16,641	7,225	32,186	859,000	32,768

We adopt geometry-amortized training (Zhou et al., 2026) by resampling a uniform subset of mesh points at each optimization step. The subset size is drawn uniformly from the integer interval specified in Table 3, and points are sampled without replacement. The lower bound is $\mathrm{round}(0.4N)$ for Elasticity, Plasticity, Airfoil, Pipe, and ShapeNet-Car, and $\mathrm{round}(0.3N)$ for SuperWing, where N is the full-mesh point count. DrivAerML uses the absolute range $[85{,}900, 859{,}000]$ for both its pressure and wall-shear-stress models. Each model remains frozen during sampling-policy evaluation.

DrivAerML uses a global batch size of four and a peak learning rate of 2×10^{-3}. Training minimizes relative L_2 error, while ShapeNet-Car follows the training objective of Transolver (Wu et al., 2024).

Inference timing. Latency is measured with batch size one after warm-up, synchronizing GPU execution before and after each request. We average 20 repetitions per case (five for DrivAerML), including the transfer of sampled predictions to the CPU. Model loading and reusable preprocessing are outside the timed region.

B.5 Inverse-Design Settings

In Figures 3 and 4, full-mesh and AdaSolver optimization share the initial geometry, flow condition, constraints, and frozen operator. Independent CFD validates the surrogate-designed geometries.

Two-dimensional airfoil. Following the parameterization of the NACA airfoil of Geo-FNO (Li et al., 2023a), seven spline-node displacements control the section and its body-fitted mesh 221×51. The initial control vector is $(0.005, 0.005, 0, 0, 0.005, 0.005, 0)$, with each component bounded by ±0.015 in chord units. At Mach 0.8 and zero incidence, SLSQP minimizes the predicted $(C_D/C_L)^2$ subject to $C_L \geq 0.05$. We use exact automatic-differentiation gradients, a maximum of 120 iterations, and a function tolerance of 10^{-9}. We restrict the optimized airfoil to remain close to the reference geometry, with thickness constrained to 80%–120% of the undeformed profile, cross-sectional area to 90%–120%, and absolute camber below 0.05 chord. We further enforce mesh validity by requiring every cell to preserve its orientation and at least 20% of its original area. AdaSolver retains 2,113 of 11,271 points (18.75%). Its initialization uses 564 coarse points, including all 121 ordered surface nodes.

Three-dimensional SuperWing. We perform aerodynamic shape optimization on SuperWing (Yang et al., 2025) under a fixed transonic flow condition. The design space contains 12 variables controlling wing sweep, dihedral, aspect and taper ratios, kink location, section thickness, and twist. We use CMA-ES (Hansen & Ostermeier, 2001) for 150 generations with 12 candidates per generation.

The optimization minimizes drag while maintaining the lift of the initial design:

$$J = \left(\frac{C_D}{0.05}\right)^2 + 200\left(C_L - C_L^{\text{target}}\right)^2 + P_{\text{geom}}, \tag{B.15}$$

where C_L^{target} is the full-mesh surrogate prediction for the initial wing. The geometry penalty P_{geom} regularizes large parameter deviations, proximity to design-space boundaries, mesh degradation, and geometric constraint violations.

AdaSolver evaluates $12{,}288$ of the $32{,}768$ surface points (37.5%). We use the normal-variation-based sampling policy described in Appendix C, and refresh the sampling density every 20 generations to adapt to the evolving wing geometry.

Independent CFD and timing. Independent CFD validation uses the CFD solver provided in the Geo-FNO (Li et al., 2023a) codebase for Airfoil and the official ADflow-based CFD pipeline released with SuperWing (Yang et al., 2025) for the wing experiments.

C Density Policy Design

This section details the sampling policies used in Tables 1 and 2. Although the refinement signals are task dependent, all policies follow the same pipeline: construct a refinement score, convert it into a target sampling density, sample the prescribed number of points, and estimate the effective density represented by the selected samples for measure correction.

Target sampling density. Let

$$\rho = \frac{1}{N}\sum_{i=1}^{N} \delta_{x_i}$$

denote the reference point measure and let n be the inference sampling budget. Each policy produces a nonnegative refinement score $r_\psi(x)$ and a corresponding positive sampling weight $t_\psi(x)$. We normalize this weight with respect to the reference measure,

$$\lambda_{\mathrm{tar}}(x) := \frac{t_\psi(x)}{\int_\Omega t_\psi(y)\,\mathrm{d}\rho(y)}, \qquad \mathrm{d}\widetilde{\rho}_{\mathrm{tar}}(x) = \lambda_{\mathrm{tar}}(x)\,\mathrm{d}\rho(x). \tag{C.1}$$

Thus, λ_{tar} directly specifies where the inference samples should be concentrated. The additive construction $t_\psi = b_0 + a_0 r_\psi$ is equivalent, after normalization, to the mixture formulation in equation 13.

Coverage-based measure correction. The target density specifies how points are sampled, while the selected samples may represent unequal portions of the reference discretization. We therefore estimate the effective coverage of each retained point and use its inverse density for the measure correction in Physics-Attention. If a selected point x_i represents m_i reference points, its correction weight is

$$c_i = \frac{m_i}{N/n}. \tag{C.2}$$

For region-based sampling, the same principle gives $c_i \propto N_k/n_k$, where N_k and n_k are the reference and retained point counts in region P_k. A common scale in c_i cancels in the self-normalized aggregation.

Shared refinement features. The three policies differ primarily in how they construct r_ψ. We use local geometric or physical priors for PI, variation in a coarse prediction for CPG, and learned intermediate features for CA. For solution-dependent policies, local variation is estimated using neighbor differences, local-mean residuals, or least-squares gradients. When the score is available only on a coarse set, it is transferred to the reference discretization by nearest-neighbor, inverse-distance, or linear interpolation. Surface signals can additionally be extended to nearby volume points with distance-decaying weights. For compactness, we summarize the shared refinement and transfer operators in Table 4. These primitives are combined differently across tasks, while all policies ultimately construct a target density λ_{tar}.

Sampling procedures. Depending on the geometry, the target density is realized either by direct weighted sampling or by regional sampling. The latter partitions the domain into spatial regions, distributes the sampling budget according to regional scores, and places points within each region using farthest-point sampling. For CPG and CA, the coarse samples S_0 are retained; only the remaining budget is sampled from the residual target density after accounting for the coverage already provided by S_0. The following subsections specify the task-specific refinement scores and density constructions.

C.1 Physics-Informed Density Refinement

PI constructs the refinement score directly from geometry or known input quantities available before inference. Each benchmark uses a simple prior that emphasizes regions where stronger solution variation or larger contributions to the target quantity are expected.

For near-wall refinement, we use the generic density

$$\lambda_{\mathrm{shell}}(x) = (1-\gamma) + \gamma\frac{\mathbf{1}_{H_d}(x)}{\rho(H_d)}, \tag{C.3}$$

Table 4: Shared primitives used to construct the sampling densities.

Notation	Definition	Role
$D_k[f]_i$	$\frac{1}{k}\sum_{j\in\mathcal{N}_k(i)}\frac{\|f_i-f_j\|_2^2}{\|z_i-z_j\|_2^2+\epsilon}$	Local gradient-energy score
$L_k[f]_i$	$\left\|f_i-\frac{1}{k}\sum_{j\in\mathcal{N}_k(i)}f_j\right\|_2$	Local curvature / graph-Laplacian proxy
$G_k[f]$	$\|\nabla_{\mathrm{LSQ}}f\|_F$	Least-squares gradient magnitude
$\mathcal{Q}_q[f]$	$\frac{f}{Q_q(f)}$	Quantile-based normalization
$\mathcal{T}_{a,b}[f]$	$\mathrm{clip}_{[0,1]}\left(\frac{f-Q_a(f)}{Q_b(f)-Q_a(f)}\right)$	Robust rescaling and clipping
$\mathcal{I}_0,\mathcal{I}_8,\mathcal{I}$	Nearest-neighbor, eight-neighbor inverse-distance, and linear interpolation, respectively	Transfer coarse scores to reference points
$\mathcal{E}_\ell[v](x)$	$e^{-d_\Gamma(x)/\ell}v(x_\Gamma)$	Extend a surface score to nearby volume points
$\mathcal{K}_\ell[v](x)$	$\max_{i\in\Gamma}e^{-\|x-x_i\|/\ell}\,\mathcal{Q}_{0.95}[v]_i$	Propagate localized surface importance
$\mathcal{P}_{K,\kappa}[s](x)$	$\frac{n_k}{n\mu_k},\qquad x\in P_k$	Region-based sampling density

where H_d contains the reference points within wall distance d. This maintains global coverage while increasing the sampling density near the boundary.

Elasticity. Stress concentrates around the internal cavity, so we increase sampling density near its boundary. Using the estimated cavity distance d_{cav}, we set

$$\lambda_{\mathrm{tar}}=\mathcal{N}[1+64\exp(-d_{\mathrm{cav}}/0.012)]\,. \tag{C.4}$$

The cavity distance is estimated from a 64×64 auxiliary grid.

Plasticity. Deformation is concentrated near die contact and beneath strongly curved parts of the die. We therefore combine depth from the contact surface with local die curvature:

$$\lambda_{\mathrm{tar}}=0.2+0.8\mathcal{N}\left[e^{-z/0.1}\left(1+0.5\frac{k_d}{\overline{k_d}}\right)\right], \tag{C.5}$$

where $z\in[0,1]$ is the normalized depth, $k_d=|\partial_{xx}S_d|$ is the die-curvature indicator, and $\overline{k_d}$ is its profile mean.

Airfoil. Aerodynamic forces are determined by the surface pressure, motivating denser sampling near the airfoil surface. With d_Γ denoting the distance to the surface, both field prediction and C_D/C_L use

$$\lambda_{\mathrm{tar}}^{\mathrm{field}}=\lambda_{\mathrm{tar}}^{C_D/C_L}=\mathcal{N}\left[1+20e^{-d_\Gamma/0.3}\right]. \tag{C.6}$$

All 121 surface nodes are retained, and the remaining samples are drawn from the interior according to this density.

Pipe. The no-slip boundary produces strong near-wall velocity gradients, so we refine the wall region while maintaining global coverage:

$$\lambda_{\mathrm{tar}}(x)=0.75+0.25\frac{\mathbf{1}_{H_{0.05}}(x)}{\rho(H_{0.05})}. \tag{C.7}$$

Darcy. Permeability discontinuities induce localized changes in the pressure gradient, so we increase resolution near coefficient interfaces. Let $\mathcal{I}_a$ contain nodes adjacent to a coefficient jump and define

$$d_a(x_i) = \min_{j \in \mathcal{I}_a} \frac{|i-j|}{84}.$$

The target density decays away from these interfaces:

$$\lambda_{\text{tar}} = \mathcal{N}[1 + 0.25 \exp(-d_a/(2/84))]\,. \tag{C.8}$$

ShapeNet-Car. Surface pressure and near-wall velocity gradients make the vehicle surface and nearby flow particularly informative. We combine uniform coverage with near-wall refinement:

$$\lambda_{\text{tar}}^{\text{field}} = 0.5 + 0.5\frac{\mathbf{1}_{H_{0.1}}}{\rho(H_{0.1})}, \qquad \lambda_{\text{tar}}^{C_D} = 0.625 + 0.375\frac{\mathbf{1}_{H_{0.1}}}{\rho(H_{0.1})}. \tag{C.9}$$

DrivAerML. Surface-normal variation identifies geometric detail relevant to pressure, while projected surface area measures sensitivity to pressure drag. We define

$$s_n(x_i) = \frac{1}{16} \sum_{j \in \mathcal{N}_{16}(i)} \|\mathbf{n}_i - \mathbf{n}_j\|_2^2, \qquad s_D(x_i) = A_i |n_{i,x}|,$$

and construct

$$\lambda_{\text{tar}}^{\text{field}} = \mathcal{N}[1 + 40\mathcal{T}_{0,0.99}[s_n]]\,, \qquad \lambda_{\text{tar}}^{C_D} = \mathcal{N}[1 + 40\mathcal{T}_{0,0.99}[s_D]]\,. \tag{C.10}$$

SuperWing. Rapid changes in surface normals identify curved and sharp wing features where additional resolution is useful. We compute

$$s_n = \left(\|\partial_I \mathbf{n}\|_2^2 + \|\partial_J \mathbf{n}\|_2^2\right)^{1/2}, \qquad \overline{s}_n = \mathcal{T}_{0.01,0.99}[s_n],$$

and use

$$\lambda_{\text{tar}}^{\text{field}} = \mathcal{N}[1 + 16\overline{s}_n^2], \qquad \lambda_{\text{tar}}^{C_D/C_L} = \mathcal{N}[1 + 8\overline{s}_n^{1/2}]. \tag{C.11}$$

Measure correction. When a tempered correction is used, the measure-correction weight c_i is replaced by c_i^β. Plasticity, Airfoil, DrivAerML, and SuperWing use coverage-based correction with $\beta = 1$. Elasticity and Darcy use $c_i = \lambda_{\text{tar}}(x_i)^{-1}$ with $\beta = 0.5$. For Pipe and ShapeNet-Car, let H_d^+ include both the near-wall samples and any supplementary points. We use

$$\lambda_{\text{eff}} = \mathcal{N}\left[(1-\gamma) + \gamma\frac{\mathbf{1}_{H_d^+}}{\rho(H_d)}\right], \qquad c_i = \lambda_{\text{eff}}(x_i)^{-1},$$

with $\beta = 0.75$ for Pipe and $\beta = 0.5$ for ShapeNet-Car.

C.2 Coarse-Prediction-Guided Density Refinement

CPG first evaluates the frozen operator on a small uniform coarse sample S_0, typically containing about 10% of the reference points. The resulting draft prediction $\widehat{u}^{(0)}$ provides a solution-dependent refinement signal. Unless stated otherwise, S_0 is retained, the remaining budget is sampled according to the resulting target density, and coverage-based measure correction uses $\beta = 1$.

Elasticity. Regions with rapidly varying stress require higher resolution. We estimate this variation from the coarse von Mises stress,

$$s = \mathcal{I}_8 D_8[\widehat{\sigma}_{\text{vM}}^{(0)}],$$

and use

$$\lambda_{\text{tar}} = 0.375 + 0.625\mathcal{N}[\min(s, Q_{0.9}(s))]\,. \tag{C.12}$$

The density is realized over 32 spatial regions with density contrast $\kappa = 4$ and farthest-point placement.

Plasticity. Localized deformation is characterized by both displacement gradients and curvature over the loading trajectory. From the coarse prediction, we compute gradient and Laplacian magnitudes $G = \|\nabla \widehat{\mathbf{u}}^{(0)}\|_F$ and $L = \|\Delta \widehat{\mathbf{u}}^{(0)}\|_2$, and combine them as

$$s = \mathcal{I}_0\left[\frac{G}{2\overline{G}_0} + \frac{L}{2\overline{L}_0}\right], \qquad \lambda_{\text{tar}} = \mathcal{P}_{24,4}[0.2 + 0.8\mathcal{N}[s]]. \tag{C.13}$$

Airfoil. For field prediction, pressure gradients identify regions requiring higher resolution; for C_D/C_L, the sampling density instead emphasizes surface regions that strongly influence drag and lift. For the field, we use

$$s_f = \mathcal{I}_0 G_{16}[\widehat{p}^{(0)}].$$

For the aerodynamic ratio, let

$$\widehat{D} = \sum_{\Gamma} b_D \widehat{p}^{(0)}, \qquad \widehat{L} = \sum_{\Gamma} b_L \widehat{p}^{(0)}$$

be the coarse drag and lift force sums. We define

$$v_R = \frac{|\widehat{L} b_D - \widehat{D} b_L|}{\max(|\widehat{L}|, 0.02)^2}, \qquad v_B = \mathcal{Q}_{0.95}[(b_L \widehat{p}^{(0)})_+] + \mathcal{Q}_{0.95}[(-b_L \widehat{p}^{(0)})_+]. \tag{C.14}$$

With $\chi = \min(1, |\widehat{L}|/0.02)$, the target densities are

$$\begin{aligned} \lambda_{\text{tar}}^{\text{field}} &= \mathcal{N}\left[(0.1 + \mathcal{Q}_{0.95}[s_f])^2\right], \\ \lambda_{\text{tar}}^{C_D/C_L} &= \mathcal{N}\left[\left(0.1 + \chi \mathcal{K}_{0.1}[v_R] + \mathcal{K}_{0.1}[|b_L|] + 2\mathcal{K}_{0.1}[v_B] + 0.5e^{-d_\Gamma/0.1}\right)^{1/2}\right]. \end{aligned} \tag{C.15}$$

All 121 surface nodes are retained and only the interior is resampled.

Pipe. High velocity variation indicates where additional flow resolution is useful. We therefore use the variation in the coarse horizontal velocity directly:

$$\lambda_{\text{tar}} = \mathcal{P}_{24,4}[D_8[\widehat{v}_x^{(0)}]]. \tag{C.16}$$

Darcy. Both pressure gradients and curvature identify localized responses to heterogeneous permeability. We combine the two signals as

$$s_{\text{Darcy}} = D_8[\widehat{p}^{(0)}] + |\Delta_{\text{FD}} \mathcal{I}_0 \widehat{p}^{(0)}|_{S_0}, \qquad \lambda_{\text{tar}} = \mathcal{P}_{32,2}[s_{\text{Darcy}}]. \tag{C.17}$$

ShapeNet-Car. Pressure variation guides field reconstruction, while pressure-force contributions identify regions relevant to drag. We construct

$$s_f = \mathcal{E}_{0.1}[D_4[\widehat{p}^{(0)}]], \qquad s_D = \mathcal{E}_{0.1}[(b_D(\widehat{p}_\Gamma^{(0)} - \overline{p}_0))_+], \tag{C.18}$$

where $\widehat{p}_\Gamma^{(0)}$ is the coarse pressure transferred to the full surface and $\overline{p}_0$ is the mean coarse surface pressure. For either score, define

$$s' = \max(\mathcal{Q}_{0.95}[s], 0.1), \qquad z_s = \text{clip}_{[0,1]}\left(\frac{s'}{Q_{0.95}(s'|_\Gamma)}\right).$$

The target density is

$$\lambda_{\text{tar}}^q = \mathcal{N}\left[(1 + A_q e^{-d_\Gamma/0.1})(1 + 2z_{s_q})\right], \qquad (A_f, A_D) = (64, 32), \quad q \in \{f, D\}. \tag{C.19}$$

The vehicle surface is retained and the remaining volume samples are selected by stratified regional sampling. Regional coverage correction uses $\beta = 1$ for field prediction and $\beta = 0.75$ for drag.

DrivAerML. Local pressure variation identifies regions important for field reconstruction, while projected area weights their relevance to drag. From

$$s = \mathcal{I}_0 L_{16}[\widehat{p}^{(0)}],$$

we define

$$\lambda_{\text{tar}}^{\text{field}} = \mathcal{N}[1 + 40\mathcal{T}_{0,0.99}[s]], \qquad \lambda_{\text{tar}}^{C_D} = \mathcal{N}[1 + 40\mathcal{T}_{0,0.99}[A|n_x|s]]. \tag{C.20}$$

Table 5: Refinement heads used by cache-aware learned refinement. MLP heads use GELU activations between linear layers.

Task	Predicted refinement quantity	Head architecture
Elasticity	von Mises stress	$128 \to 4 \to 1$
Plasticity	Aggregated displacement-gradient magnitude	$128 \to 4 \to 32 \to 1$
Airfoil	Pressure	$128 \to 8 \to 32 \to 1$
Pipe	Velocity-gradient magnitude	2 Transolver blocks, width 128
Darcy	CPG refinement score s_{Darcy}	2 Transolver blocks, width 128
ShapeNet-Car	Surface pressure	$256 \to 64 \to 1$
DrivAerML	Local pressure-variation score	$256 \to 8 \to 32 \to 1$
SuperWing	Joint pressure–friction curvature score	$128 \to 4 \to 1$

SuperWing. Localized aerodynamic structures are captured by curvature in the predicted pressure and skin-friction fields. With $\mathbf{u} = (C_p, 150C_{f,\tau}, 300C_{f,z})$, we compute

$$s = \mathcal{T}_{0.01,0.99}\left[\|\Delta_5 \mathcal{I}\widehat{\mathbf{u}}^{(0)}\|_2\right].$$

For C_D/C_L, this solution-dependent signal is blended with the geometric normal-variation prior $g = \overline{s}_n$:

$$\begin{aligned} d_\omega &= (1-\omega)\mathcal{N}[1+8g^{1/2}] + \omega\mathcal{N}[1+8s^{1/2}], \\ \mathcal{B}_\omega[g,s] &= \mathcal{N}\left[\max\left(d_\omega, \frac{\max_x d_\omega(x)}{8}\right)\right]. \end{aligned} \tag{C.21}$$

The final densities are

$$\lambda_{\mathrm{tar}}^{\mathrm{field}} = \mathcal{P}_{64,4}[s], \qquad \lambda_{\mathrm{tar}}^{C_D/C_L} = \mathcal{B}_{0.75}[g,s]. \tag{C.22}$$

The C_D/C_L policy uses the residual-sampling construction described above.

C.3 Cache-Aware Learned Refinement

CA replaces the complete coarse prediction used by CPG with a learned refinement signal extracted from early frozen-backbone features. For each task, an auxiliary refinement head is trained offline while keeping the backbone fixed. The predicted signal is converted into a target density using the same sampling machinery as CPG. Continuous field targets are standardized, while nonnegative score targets are transformed with $\log(1+s)$ before standardization. The head architectures and prediction targets are summarized in Table 5.

Elasticity. The learned signal should identify regions of large stress variation. The head predicts the von Mises stress, from which we compute

$$s = \mathcal{I}_8 D_8[\widehat{\sigma}_{\mathrm{vM}}^{\mathrm{head}}].$$

Using regional maxima $v_k = \max_{x \in P_k} s(x)$, we define

$$\lambda_{\mathrm{tar}}(x) = 0.75 + 0.25\frac{\min(v_k, Q_{0.99}(v))^2}{\sum_j \mu_j \min(v_j, Q_{0.99}(v))^2}, \qquad x \in P_k. \tag{C.23}$$

Plasticity. The displacement-gradient magnitude provides a compact indicator of localized deformation over the full loading trajectory. The head learns

$$G_{\mathrm{GT}} = \left(\sum_{t,c} \|\nabla u_{t,c}\|_2^2\right)^{1/2},$$

and its prediction gives

$$\lambda_{\mathrm{tar}} = 0.2 + 0.8\mathcal{N}\left[\mathcal{I}_0 \widehat{G}^{\mathrm{head}}\right]. \tag{C.24}$$

Airfoil. Pressure gradients provide the learned refinement signal for both field prediction and the aerodynamic ratio. From the head prediction, we define

$$s_q = \mathcal{I}_0 G_{16}[\widehat{p}^{\mathrm{head},q}], \qquad q \in \{\mathrm{field}, C_D/C_L\}.$$

The corresponding densities are

$$\begin{aligned} \lambda_{\mathrm{tar}}^{\mathrm{field}} &= \mathcal{N}\left[(0.01 + \mathcal{Q}_{0.95}[s_{\mathrm{field}}])^2\right], \\ \lambda_{\mathrm{tar}}^{C_D/C_L} &= \mathcal{N}\left[(0.1 + \mathcal{Q}_{0.95}[s_{C_D/C_L}])^{1/2}\right]. \end{aligned} \tag{C.25}$$

The coarse sample retains all 121 surface nodes. Policy design follows the $C_L > 0.01$ inclusion rule, while test evaluation uses $C_L > 0.02$.

Pipe. Velocity-gradient magnitude directly identifies regions requiring additional flow resolution. The refinement head predicts this quantity, and we use

$$\lambda_{\mathrm{tar}} = \mathcal{P}_{24,4}[(\widehat{G}^{\mathrm{head}})^2]. \tag{C.26}$$

Darcy. We directly learn the CPG pressure-variation score, avoiding an explicit coarse solution while preserving its refinement signal:

$$\lambda_{\mathrm{tar}} = \mathcal{P}_{32,2}[\widehat{s}^{\mathrm{head}}_{\mathrm{Darcy}}]. \tag{C.27}$$

ShapeNet-Car. The learned pressure estimate provides the same local-variation and drag-sensitive signals used by CPG without requiring a complete coarse forward pass. Replacing $\widehat{p}^{(0)}$ with $\widehat{p}^{\mathrm{head}}$ in the CPG score gives

$$\lambda_{\mathrm{tar}}^{q} = \mathcal{N}\left[(1 + A_q e^{-d_\Gamma/0.1})(1 + 2z_{s_q^{\mathrm{head}}})\right], \qquad (A_f, A_D) = (64, 32), \quad q \in \{f, D\}. \tag{C.28}$$

Surface retention, regional volume sampling, and measure correction follow the corresponding CPG policy.

DrivAerML. The refinement head learns the local pressure-variation signal used by CPG. After transferring its prediction,

$$s = \mathcal{I}_0 \widehat{L}^{\mathrm{head}},$$

we use

$$\lambda_{\mathrm{tar}}^{\mathrm{field}} = \mathcal{N}\left[1 + 512\mathcal{T}_{0,0.99}[s]^2\right], \qquad \lambda_{\mathrm{tar}}^{C_D} = \mathcal{N}[1 + 40\mathcal{T}_{0,0.99}[A|n_x|s]]. \tag{C.29}$$

SuperWing. The learned score approximates curvature in the joint pressure and skin-friction fields, identifying localized aerodynamic structures. We define

$$s = \mathcal{T}_{0.01,0.99}[\mathcal{I}\widehat{s}^{\mathrm{head}}],$$

and use

$$\lambda_{\mathrm{tar}}^{\mathrm{field}} = \mathcal{P}_{64,4}[s^{1/2}], \qquad \lambda_{\mathrm{tar}}^{C_D/C_L} = \mathcal{B}_{0.5}[g, s]. \tag{C.30}$$

The C_D/C_L policy combines the learned signal with the same geometric prior used by CPG.

D ADDITIONAL RESULTS

D.1 ABLATION OF MEASURE-AWARE PHYSICS-ATTENTION

Section 3.2 shows that changing the sampling density at inference changes the point-to-slice aggregation in Physics-Attention and therefore introduces a sampling-measure bias unless the aggregation is properly compensated. The adaptive policies in our main experiments use policy-specific measure-correction weights $c_i^{\beta^\star}$, where c_i is determined from the effective coverage of the selected samples and $\beta^\star$ is the correction exponent specified during policy design. Here we directly test how sensitive inference is to a *mismatch* in this compensation.

For a fixed set of sampled points, the measure-aware physics token is formed as

$$\widehat{u}_{s,j}^{(\beta)} = \frac{\sum_i w_{i,j} c_i^\beta h_i}{\sum_i w_{i,j} c_i^\beta}, \tag{D.1}$$

where β controls the strength of the density compensation. In particular, the relative contrast of two correction weights satisfies

$$\log \frac{c_i^\beta}{c_k^\beta} = \beta \log \frac{c_i}{c_k}. \tag{D.2}$$

Thus, $\beta = 0$ removes density compensation entirely and recovers the original Transolver point-to-slice aggregation, whereas increasing β strengthens the density-dependent reweighting.

Importantly, the reference point in this ablation is the policy-specific exponent $\beta^\star$ used in the main experiments, which is not necessarily equal to one. Relative to this deployed correction, changing β modifies the pairwise compensation ratio as

$$\frac{c_i^\beta / c_k^\beta}{c_i^{\beta^\star} / c_k^{\beta^\star}} = \left(\frac{c_i}{c_k}\right)^{\beta - \beta^\star}. \tag{D.3}$$

Hence, $\beta < \beta^\star$ under-compensates the sampling-induced density shift relative to the main policy, while $\beta > \beta^\star$ applies a stronger compensation than that used by the policy.

We isolate this effect from adaptive sampling itself by keeping the sampling realization fixed. For each PI, CPG, and CA policy, we first generate the coarse and final point sets exactly as in the main experiments using the same policy, budget, and sampling seed. We then rerun the frozen Transolver on these *same sampled points*, changing only the exponent β in the inference-time compensation weights c_i^β. Operator and refinement-head checkpoints, coarse and final sampled indices, reconstruction, and test examples are unchanged. We evaluate $\beta \in \{0, 0.25, 0.5, 0.75, 1\}$ at retained fractions of 37.5%, 50%, and 75%, using sampling seeds 0, 1, and 2.

Let $A_{m,s}(\beta)$ denote the trapezoidal area under the error–budget curve for policy m and sampling seed s. Figure 5 reports

$$R_m(\beta) = \frac{1}{3} \sum_{s=0}^{2} \frac{A_{m,s}(\beta)}{A_{m,s}(0)}. \tag{D.4}$$

Normalization is performed separately for each policy and seed before averaging. Therefore, $R_m(0) = 1$ corresponds to the same adaptive samples evaluated by the original Transolver without measure correction, and $R_m(\beta) < 1$ indicates that inference-time compensation reduces the error–budget area. The marker at $\beta^\star$ denotes the correction exponent used by that policy in the main experiments. Because the sampled locations are identical along each curve, changes in $R_m(\beta)$ measure only the effect of changing the inference-time aggregation weights.

The results show a clear sensitivity to compensation mismatch. Across most datasets and policies, the lowest or near-lowest error is obtained when the inference-time correction is matched to the policy-specific $\beta^\star$, while moving away from this setting increases the error–budget area. This behavior is consistent with the sampling-measure analysis: once the spatial samples are fixed, changing $c_i^{\beta^\star}$ to c_i^β changes the relative contribution of the same point features to the physics tokens, introducing an aggregation bias that propagates through the frozen operator.

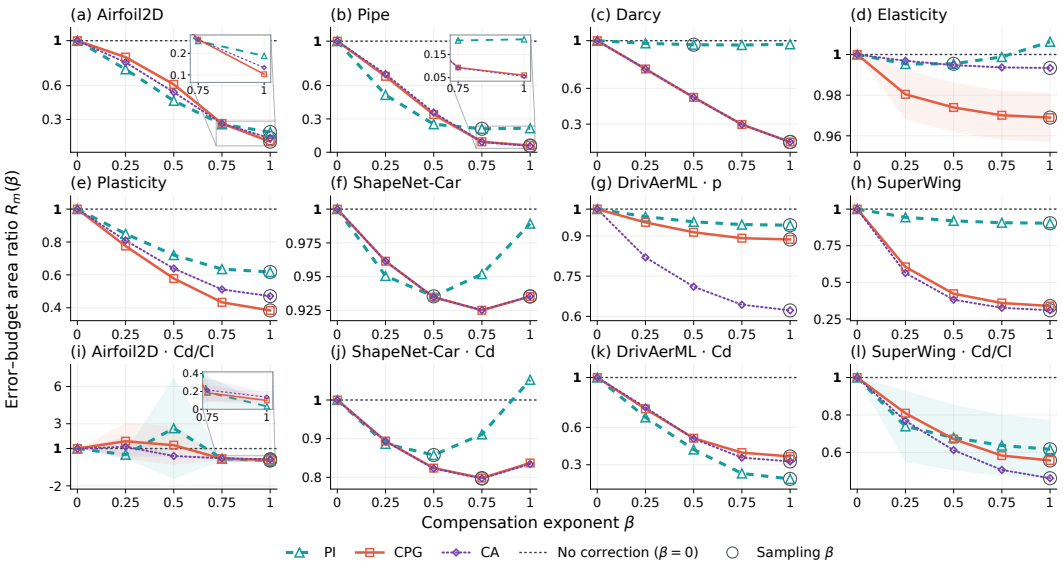

Figure 5: **Ablation of the measure-correction exponent.** Checkpoints, sampled points, and reconstruction are fixed. The y-axis is $R_m(\beta)$ from equation D.4: each policy and seed is normalized to its own $\beta = 0$ error–budget area before averaging. Areas cover 37.5%, 50%, and 75% budgets. Bands show one sample standard deviation across three seeds. Gray rings (Sampling β) mark the compensation exponent used by each policy in the main experiments. The dashed line denotes no correction. Insets enlarge high-β portions; the Airfoil ratio panel retains the full uncertainty envelope. Table 6 uses Uniform-normalized rAUEC.

The effect can be substantial. For CPG, removing the correction ($\beta = 0$) increases the error–budget area by a factor of approximately 9.7 relative to full correction on Airfoil field prediction and by approximately 16.4 on Pipe. On Darcy, the CPG and CA rAUECs change from 0.642 and 0.640 at their main-policy settings to 4.278 and 4.265 without compensation, despite using exactly the same sampled locations. Similarly, the main correction reduces the SuperWing field error–budget area by 66.1% for CPG and 69.1% for CA, and reduces the DrivAerML drag area by 63.3% and 67.3%, respectively.

Table 6: Measure correction on fixed adaptive samples. Entries are mean rAUEC over three sampling seeds, with Uniform equal to one. Bold marks the lower value in each within-policy $\beta = 0/\beta^\star$ pair. Parentheses give the percentage decrease in mean rAUEC from $\beta = 0$, computed before rounding. The compensation exponents $\beta^\star$ are specified in Appendix C.

Dataset	Metric	PI		CPG		CA	
		$\beta = 0$	$\beta^\star$	$\beta = 0$	$\beta^\star$	$\beta = 0$	$\beta^\star$
Airfoil	Field	3.2852	**0.6166** (81.2%)	5.6733	**0.5831** (89.7%)	4.1829	**0.5579** (86.7%)
Pipe	Field	4.1234	**0.8711** (78.9%)	11.1004	**0.6719** (93.9%)	12.7244	**0.7051** (94.5%)
Darcy	Field	1.0315	**0.9965** (3.4%)	4.2778	**0.6419** (85.0%)	4.2654	**0.6400** (85.0%)
Elasticity	Field	0.7314	**0.7280** (0.5%)	0.4832	**0.4681** (3.1%)	0.5506	**0.5469** (0.7%)
Plasticity	Field	1.4393	**0.8888** (38.2%)	2.3153	**0.8890** (61.6%)	1.8908	**0.8897** (52.9%)
ShapeNet-Car	Field	0.6301	**0.5895** (6.4%)	0.5962	**0.5577** (6.5%)	0.5962	**0.5577** (6.4%)
DrivAerML	Field	0.9398	**0.8834** (6.0%)	0.9298	**0.8246** (11.3%)	1.3280	**0.8263** (37.8%)
SuperWing	Field	0.8369	**0.7564** (9.6%)	1.9796	**0.6706** (66.1%)	2.3252	**0.7193** (69.1%)
Airfoil	Cd/Cl	11.3964	**0.2174** (98.1%)	2.1147	**0.2379** (88.7%)	1.7583	**0.2453** (86.1%)
ShapeNet-Car	Cd	0.7649	**0.6561** (14.2%)	0.8113	**0.6474** (20.2%)	0.8130	**0.6479** (20.3%)
DrivAerML	Cd	4.9553	**0.9245** (81.3%)	2.5121	**0.9209** (63.3%)	2.8282	**0.9244** (67.3%)
SuperWing	Cd/Cl	1.4093	**0.8246** (41.5%)	1.4403	**0.8026** (44.3%)	1.7669	**0.8167** (53.8%)

The non-unit optimal exponents provide an especially informative control. For example, on ShapeNet-Car drag prediction, CPG and CA attain their lowest areas around the policy value $\beta^\star = 0.75$, rather than at $\beta = 1$, while PI favors the smaller correction used by its policy. Thus, the observed improvement is not simply a consequence of applying stronger reweighting. Instead, inference is most

accurate when the compensation strength is consistent with that used to construct the corresponding adaptive policy.

Overall, this controlled experiment provides direct empirical evidence for the sampling-measure bias analyzed in Section 3.2. Adaptive sampling determines *where* the frozen operator is evaluated, while the measure correction determines *how* those sampled features are aggregated. Holding the former fixed and perturbing only the latter shows that a mismatch in compensation can substantially degrade prediction accuracy, whereas the policy-matched correction restores the intended inference behavior.

D.2 More Results

Figure 6 compares Uniform and the three AdaSolver policies at 37.5%, 50%, and 75% point budgets with full-mesh-trained Transolver references, including their 100% endpoints. Errors are divided by the full-mesh error of the backbone shared by Uniform and AdaSolver; the horizontal line at one marks this reference. Shaded bands show one sample standard deviation across three sampling seeds.

References use the original Transolver checkpoints for Pipe and Elasticity, and task-specific checkpoints trained on full meshes for the remaining tasks. Point-cloud models reuse Uniform's samples and reconstruction; Pipe uses Cartesian subgrids at the plotted retained fractions. Darcy's full-mesh-trained backbone makes its two curves coincide. DrivAerML changes the pressure checkpoint while retaining Uniform's wall-shear-stress predictions. The Airfoil reference is trained on pressure using all 11,271 points per training step, with the same architecture, 1,000 epochs, and global batch of eight as the variable-point backbone. Both Airfoil panels use the same 110 cases and the shared pressure backbone's full-mesh error for normalization. Airfoil C_D/C_L uses absolute ratio error. Its axis is linear through five and logarithmic above five, with the transition marked. Elasticity uses a logarithmic axis.

On Plasticity, CPG at 37.5% achieves 5.02% relative L_2 error, compared with 6.40% for Transolver at 100%: 62.5% fewer points and 21.6% lower error. On DrivAerML, the corresponding C_D errors are 0.948% and 1.05%. On Airfoil, CPG at 37.5% gives 0.0186 absolute C_D/C_L error, compared with 0.0281 for the full-mesh-trained reference at 100%. The benefit depends on the task and output quantity: in Darcy, CPG and CA approach full-mesh field accuracy at small budgets, while PI stays close to Uniform.

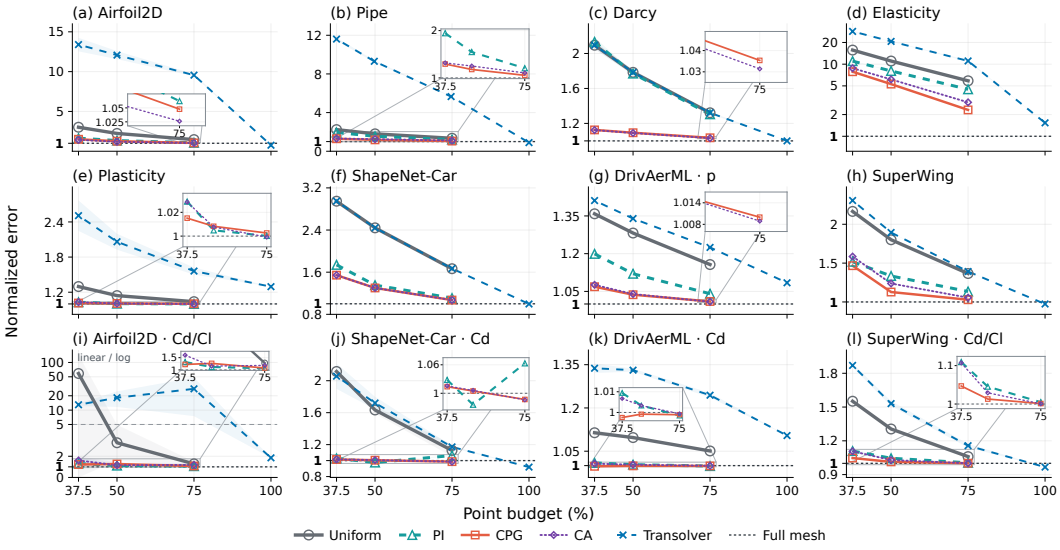

Figure 6: **Accuracy across point budgets.** Panels (a)–(h) show field metrics and (i)–(l) show aerodynamic quantities. Transolver is the full-mesh-trained reference; the horizontal line marks full-mesh accuracy of AdaSolver's backbone. Airfoil uses 110 cases with reference $C_L > 0.02$ and a pressure-trained Transolver reference. Its C_D/C_L axis is linear through five, then logarithmic. Insets enlarge mean curves.

D.3 Density Visualization

Figure 7 compares point allocation for the eight showcase cases at a 37.5% budget. Within each spatial bin B, we display the retained fraction $n_{\text{selected}}(B)/n_{\text{reference}}(B)$ on a common scale. Uniform gives approximately constant retention, while the adaptive policies redistribute points across the domain.

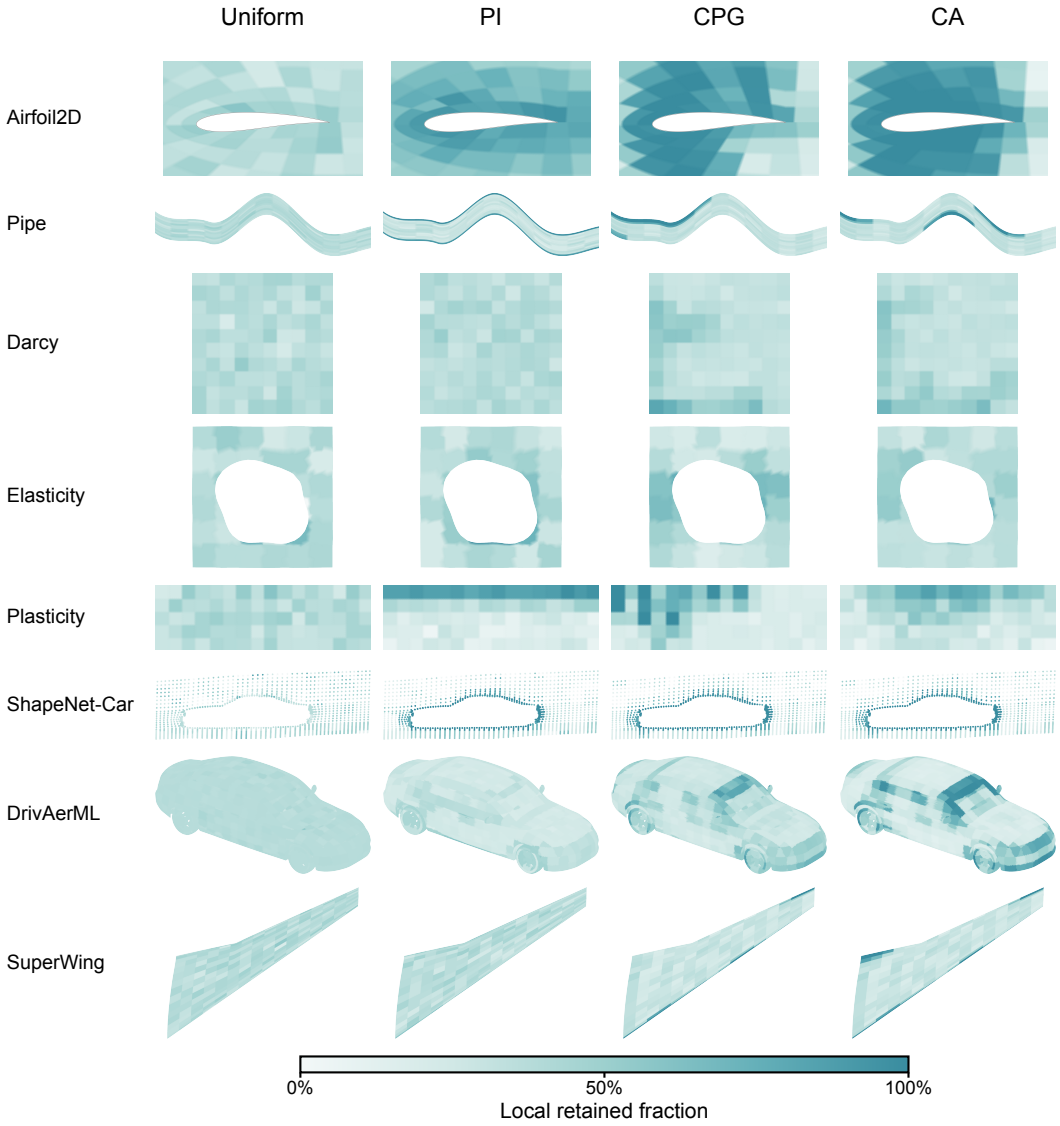

Figure 7: **Local sampling retention.** Rows show the same eight cases used in Appendix D.5, with sampling seed zero. Darker colors indicate a larger fraction of reference points retained within each bin. All methods share the same color scale.

D.4 COMPUTATIONAL COST

Adaptive inference reduces operator evaluation cost by processing fewer spatial points, but introduces additional overhead for acquiring the refinement signal. Table 7 quantifies this trade-off across all eight benchmarks at the three evaluation budgets. All methods use the same frozen backbone. Uniform provides a reference at matched point counts, while full mesh measures the cost of evaluating all reference points. The reported latency includes online refinement and prediction at the selected points, with reusable preprocessing performed in advance. Reconstruction onto the full mesh is excluded.

Task	Budget	Full mesh	Uniform	PI	CPG	CA
Airfoil2D	37.5%	5.280 ± 0.009	2.438 ± 0.018	2.539 ± 0.016	4.435 ± 0.039	3.110 ± 0.031
	50%		3.037 ± 0.005	3.176 ± 0.006	5.050 ± 0.012	3.775 ± 0.012
	75%		4.049 ± 0.009	4.218 ± 0.013	6.104 ± 0.018	4.847 ± 0.016
Pipe	37.5%	7.407 ± 0.004	3.238 ± 0.002	3.382 ± 0.004	5.067 ± 0.005	4.095 ± 0.005
	50%		4.038 ± 0.005	4.199 ± 0.005	5.891 ± 0.007	4.923 ± 0.007
	75%		5.780 ± 0.021	6.025 ± 0.016	7.721 ± 0.020	6.774 ± 0.028
Darcy	37.5%	3.525 ± 0.001	1.929 ± 0.002	1.994 ± 0.003	3.469 ± 0.007	2.675 ± 0.004
	50%		2.212 ± 0.001	2.301 ± 0.003	3.785 ± 0.017	3.002 ± 0.015
	75%		2.907 ± 0.005	3.030 ± 0.004	4.536 ± 0.014	3.738 ± 0.006
Elasticity†	37.5%	1.264 ± 0.013	0.987 ± 0.007	1.009 ± 0.006	2.086 ± 0.013	1.524 ± 0.024
	50%		1.027 ± 0.004	1.047 ± 0.003	2.209 ± 0.003	1.598 ± 0.017
	75%		1.166 ± 0.003	1.184 ± 0.004	2.401 ± 0.038	1.756 ± 0.012
Plasticity	37.5%	2.049 ± 0.006	1.259 ± 0.002	1.289 ± 0.003	2.516 ± 0.005	1.657 ± 0.003
	50%		1.430 ± 0.002	1.470 ± 0.002	2.716 ± 0.007	1.833 ± 0.004
	75%		1.708 ± 0.062	1.772 ± 0.071	3.049 ± 0.065	2.152 ± 0.078
ShapeNet-Car	37.5%	16.359 ± 0.011	6.856 ± 0.004	6.976 ± 0.005	10.609 ± 0.026	8.401 ± 0.027
	50%		8.644 ± 0.008	8.773 ± 0.006	12.404 ± 0.025	10.259 ± 0.027
	75%		12.388 ± 0.004	12.618 ± 0.006	16.258 ± 0.029	14.148 ± 0.029
DrivAerML	37.5%	4601.5 ± 140.1	1726.7 ± 46.4	1758.6 ± 47.6	2228.2 ± 61.2	1765.0 ± 46.6
	50%		2306.9 ± 62.1	2347.1 ± 62.2	2825.2 ± 74.3	2345.7 ± 59.6
	75%		3460.8 ± 97.0	3520.6 ± 97.2	4005.8 ± 109.6	3505.5 ± 94.4
SuperWing	37.5%	13.602 ± 0.010	5.671 ± 0.002	5.909 ± 0.003	8.610 ± 0.006	6.659 ± 0.007
	50%		7.401 ± 0.005	7.744 ± 0.005	10.447 ± 0.007	8.487 ± 0.006
	75%		10.483 ± 0.010	10.987 ± 0.008	13.690 ± 0.010	11.763 ± 0.010

Table 7: **Inference latency at different point budgets (ms).** Values are means $\pm$ standard deviations across test cases and repeated samplings. The full-mesh column uses all reference points and three cases, and is shown once per task. † The Elasticity timing configuration uses a 25% coarse set and a two-block head. Timings cover prediction at the selected points and online refinement, excluding full-field reconstruction.

Refinement overhead. PI remains close to Uniform, adding 1.5–4.8% latency across the 24 task–budget pairs. Its geometry-based density can be reused across predictions. CPG incurs a larger overhead because it evaluates a complete coarse prediction before selecting additional points. CA obtains its refinement signal from early features and reuses cached pointwise quantities during the final evaluation. It reduces latency by 12.3–34.1% relative to CPG, with a reduction in every measured setting. Thus, the learned refinement signal substantially reduces the runtime overhead of input-adaptive sampling compared with coarse-prediction-based refinement.

Realized speedup. At a 37.5% point budget, CA is faster than full-mesh inference on seven of the eight benchmarks. The speedups, computed as $T_{\text{full}}/T_{\text{CA}}$, reach $1.95\times$ on ShapeNet-Car, $2.61\times$ on DrivAerML, and $2.04\times$ on SuperWing. These gains are more pronounced when point-dependent operator evaluation constitutes a large fraction of the total inference cost. On Elasticity, CA takes 1.52 ms versus 1.26 ms for full mesh: refinement overhead exceeds the savings in point-dependent computation. Point reduction does not imply a proportional reduction in latency. Consequently, the realized speedup is governed by the balance between the operator computation eliminated by point reduction and the additional cost of acquiring the refinement signal.

D.5 More Showcases

The following pages connect the aggregate error profiles to spatial sampling and reconstruction. Each page shows one held-out geometry at a 37.5% point budget with sampling seed zero. These examples were selected for visible differences between sampling policies. Their annotations report errors for the displayed case. Figure 6 summarizes the test populations.

The first two rows show sampling locations and the policy's refinement feature. Features are rescaled from each policy's minimum to its 99th percentile for display, with larger values clipped at the upper end of the colour scale. Uniform has no refinement feature. Each physical quantity then has a ground-truth view and matched prediction and error rows. Field and error ranges are shared across methods within that quantity. Colour-bar arrows indicate values outside the displayed range. Component errors are signed prediction-minus-ground-truth residuals. Magnitude rows show nonnegative vector error norms, and Plasticity uses the temporal aggregation specified below.

D.5.1 Airfoil

The pressure jump in this example makes errors in its location visible as a narrow residual band. All three adaptive policies reduce the field error relative to Uniform. CA gives the lowest error for this geometry (1.30%), with PI and CPG attaining 1.33% and 1.35%, respectively.

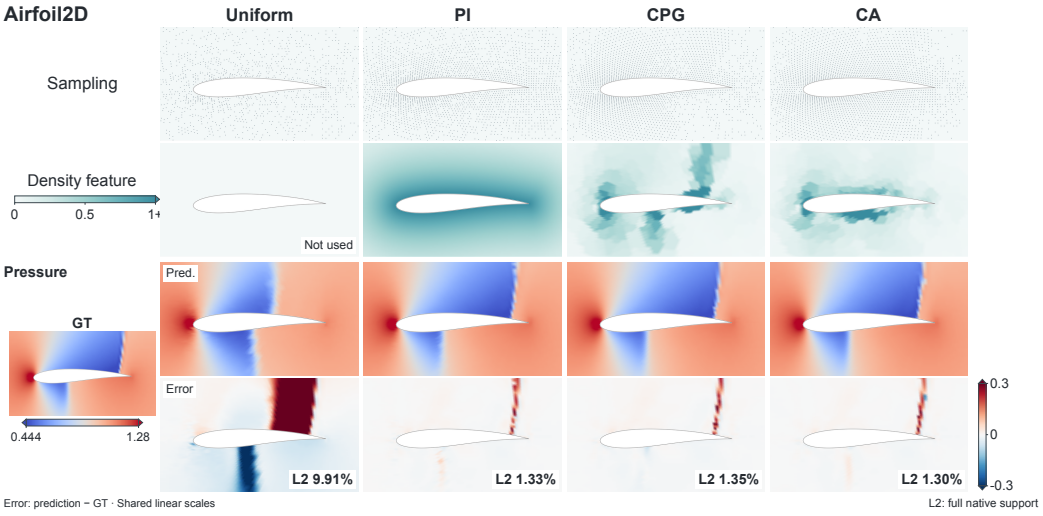

Figure 8: **Airfoil.** Sampling and pressure prediction.

D.5.2 PIPE

The predicted horizontal-velocity fields have similar overall structure, while the shared residual scale exposes differences along the curved pipe. CPG and CA reduce the error to 0.24% and 0.30%, respectively, compared with 1.31% under Uniform sampling.

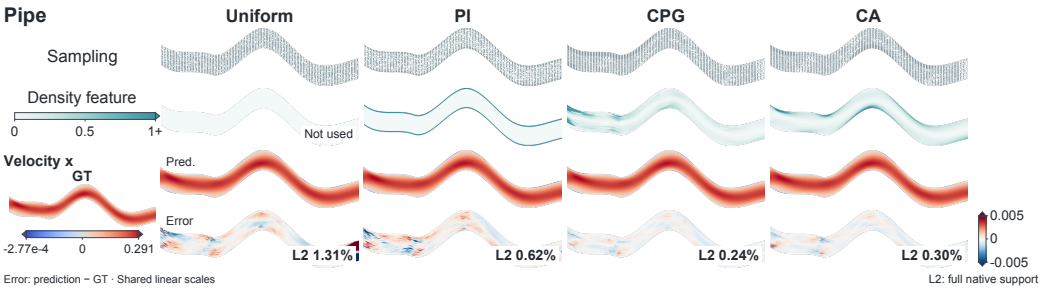

Figure 9: **Pipe.** Sampling and horizontal velocity prediction.

D.5.3 DARCY

The refinement policies assign different importance to regions of the heterogeneous medium. The CPG and CA allocations produce similar solution fields and smaller residuals than Uniform in this example, consistent with their aggregate advantage on Darcy in Figure 6.

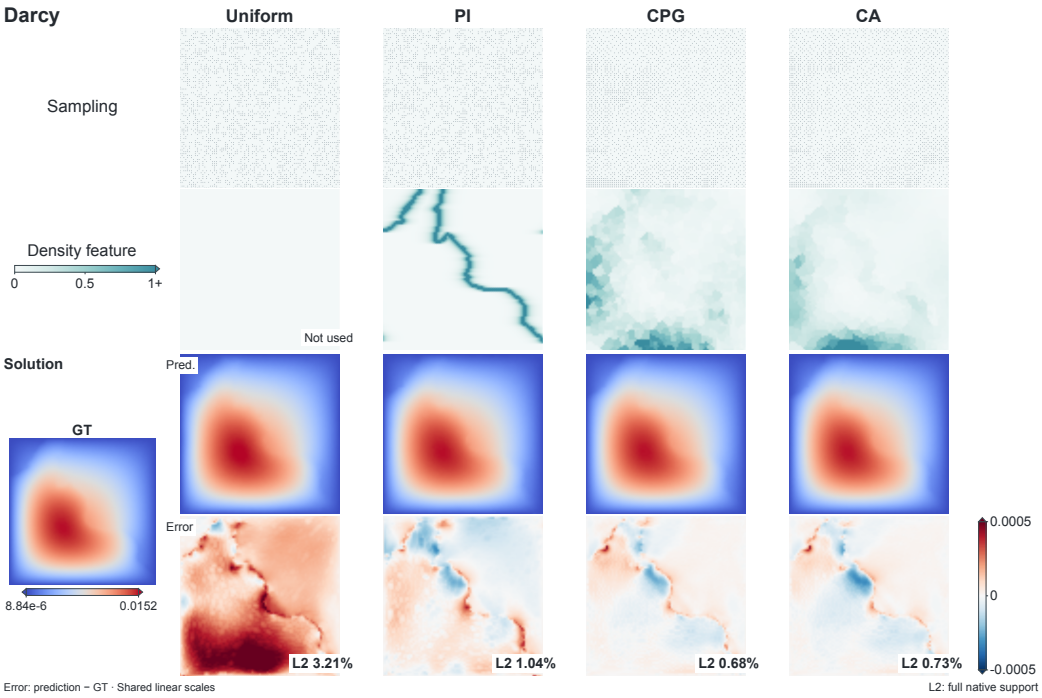

Figure 10: **Darcy.** Sampling and pressure prediction.

D.5.4 ELASTICITY

The irregular hole creates localized stress structure. The residual views show the reduction in reconstruction error around this region under adaptive sampling. CPG and CA attain similar errors, with PI also improving substantially upon the uniform allocation.

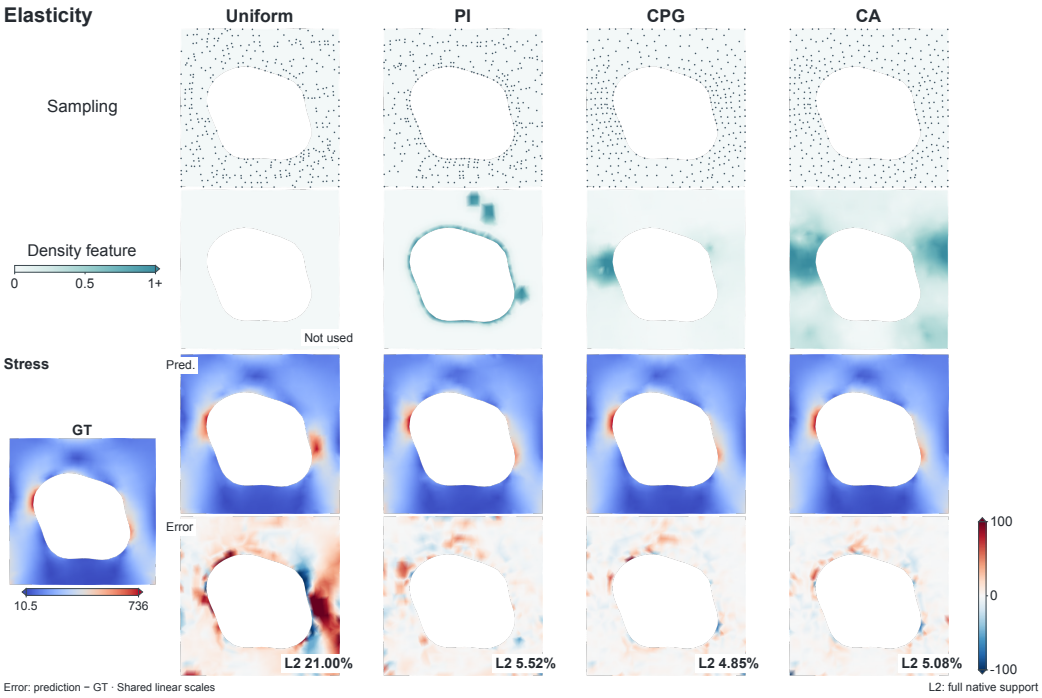

Figure 11: **Elasticity.** Sampling and stress prediction.

D.5.5 PLASTICITY

Plasticity predicts a sequence of 20 displacement fields. We summarize all recorded steps with equal weights, $w_t = 1/20$, because the time grid is uniform. The field rows show time-averaged components and displacement magnitude. Component-error rows average the absolute residual at each step. The magnitude-error row averages the vector residual norm. The annotated L_2 errors use the weighted space–time fields over all 20 steps.

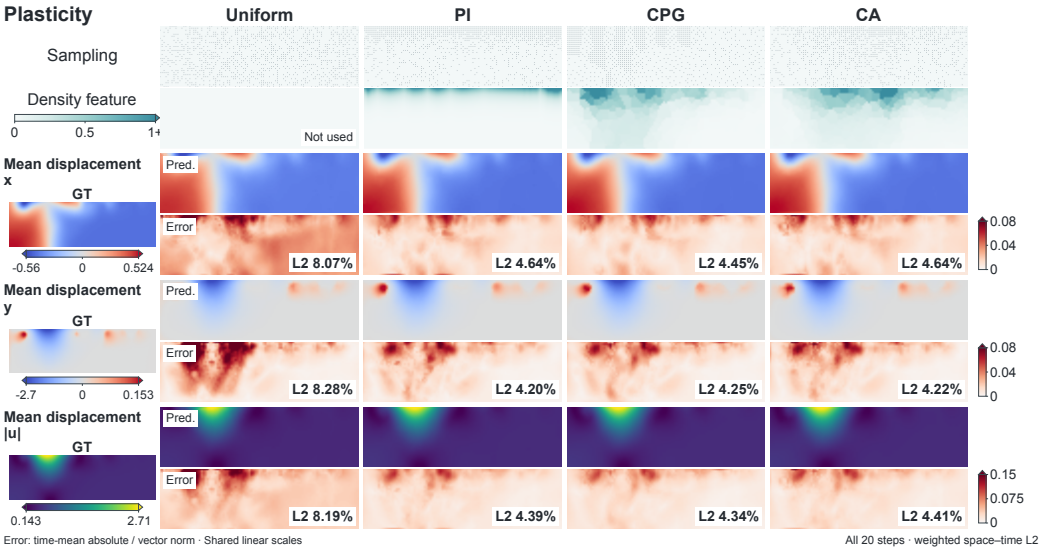

Figure 12: **Plasticity.** Sampling and displacement prediction.

D.5.6 SHAPENET-CAR

The automotive examples contain both surface and volumetric outputs. For ShapeNet-Car, pressure is shown on the surface, and a fixed geometric slice exposes the three velocity components and speed in the surrounding volume. Error annotations use each field's full native support. This view connects changes in point allocation to both pressure and velocity accuracy.

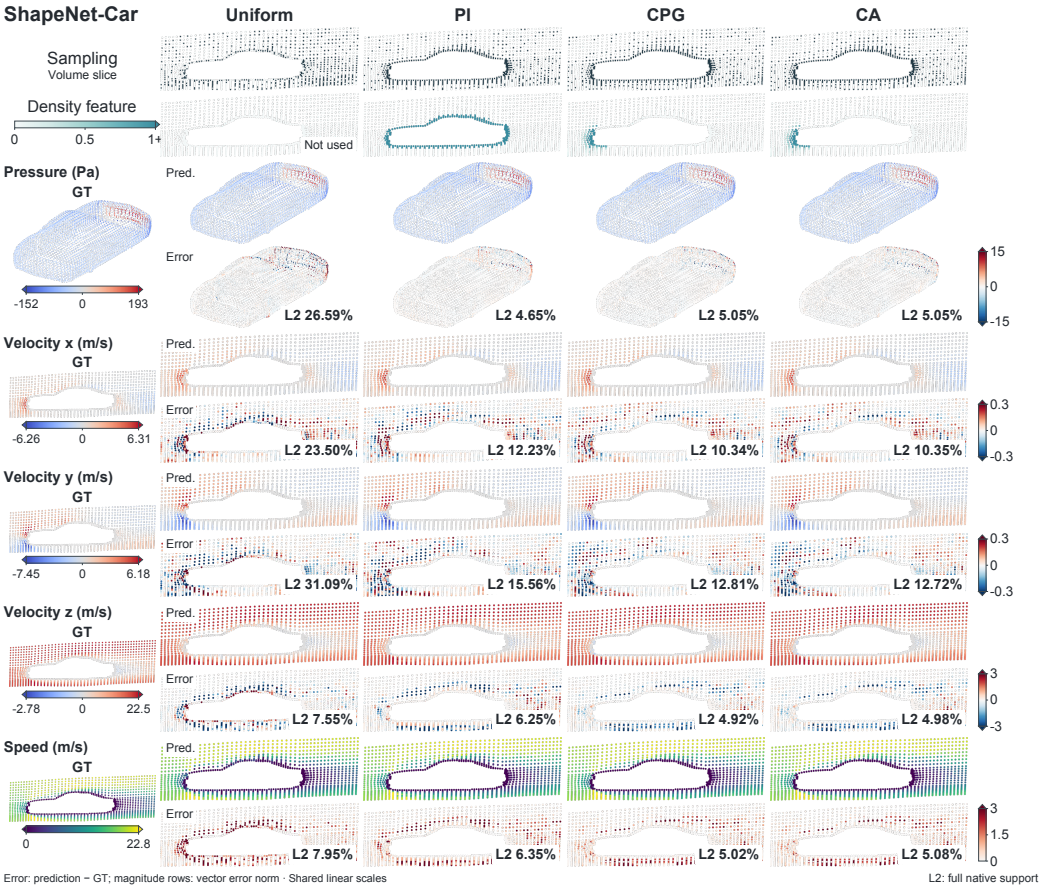

Figure 13: **ShapeNet-Car.** Sampling and prediction of surface pressure and volumetric velocity.

D.5.7 DRIVAERML

DrivAerML provides pressure and wall-shear-stress predictions on the same vehicle surface. We reuse each pressure sampler's point indices for the frozen wall-shear-stress operator. The figure therefore shows how these allocations affect pressure, all three shear components, and their vector magnitude on a shared geometry.

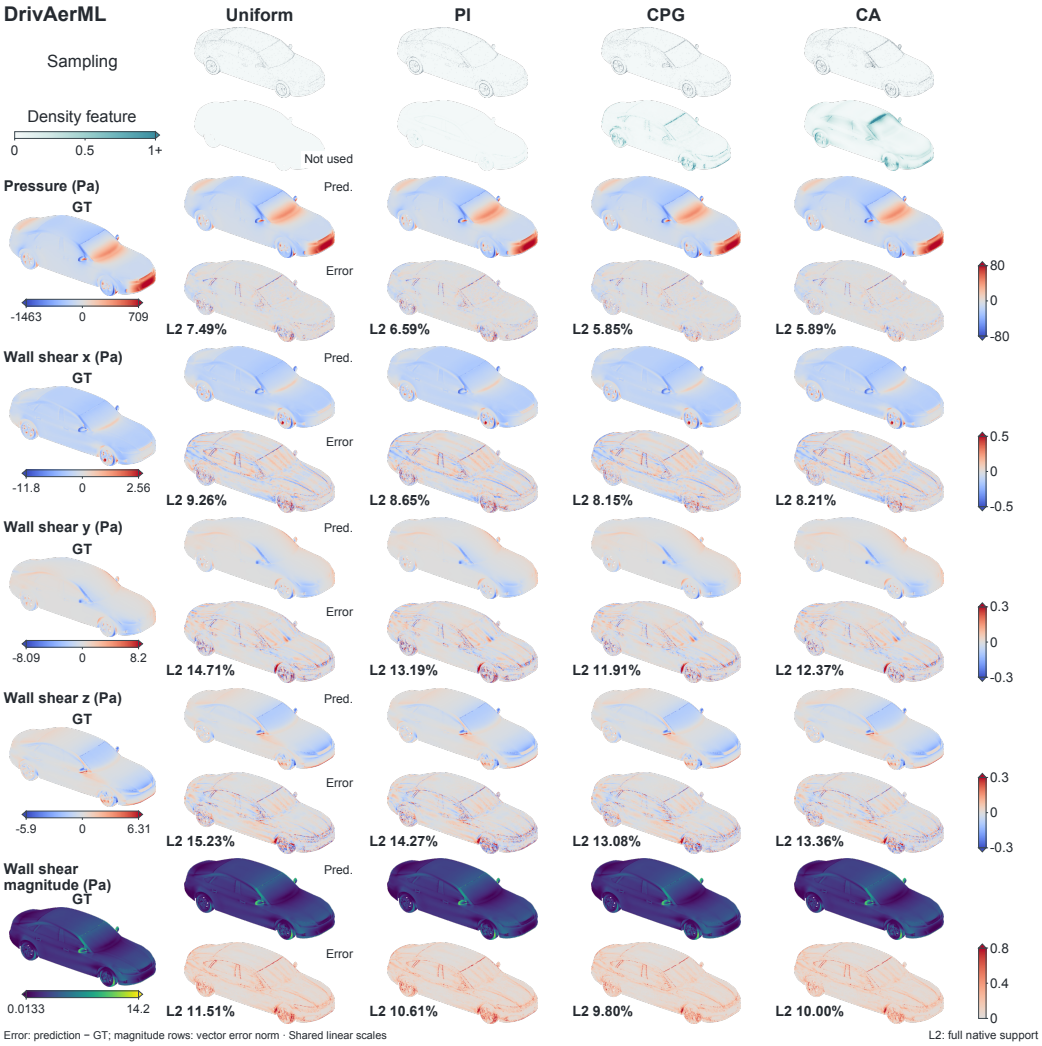

Figure 14: **DrivAerML.** Sampling and prediction of surface pressure and wall shear stress.

D.5.8 SUPERWING

The SuperWing page retains pressure and both friction components, together with friction magnitude. The stored friction-channel scaling factors are undone before visualization. Policy ordering varies across these outputs: CPG gives the smallest pressure error for this wing, while PI gives the smallest tangent-friction error. The individual rows expose this variation beyond the joint field metric.

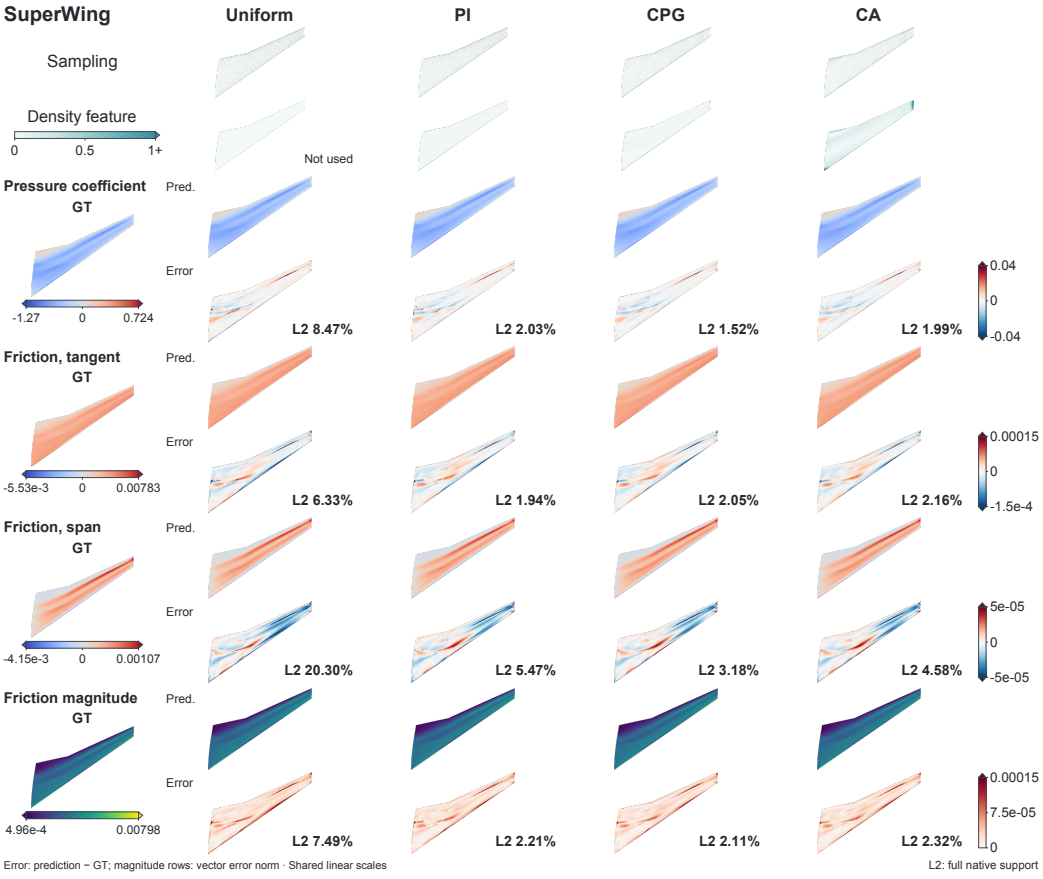

Figure 15: **SuperWing.** Sampling and prediction of pressure and skin-friction coefficients.