THRESHOLD-AWARE CONFORMAL ROUTING

Anonymous authors
Paper under double-blind review

ABSTRACT

High-fidelity simulations are essential to scientific and engineering design, but can be expensive to run repeatedly. Learned surrogates offer a faster alternative, yet their higher errors may alter downstream decisions. This accuracy-speed tradeoff creates a need to determine whether a surrogate can be used or the full simulator remains necessary. We study decisions determined by whether a scalar quantity of interest lies above or below a fixed threshold. For each input, we use the surrogate when its conformal interval lies entirely on one side of the threshold and route the input to simulation when the interval intersects it. Standard conformal prediction constructs intervals without reference to the downstream decision threshold: even a narrow interval near the threshold can cross it and trigger simulation, whereas a wider interval farther away can remain entirely on one side and require no simulation. We introduce Threshold-Aware Conformal Routing (TACR), which learns an input-dependent scale using a threshold-aware objective that concentrates interval tightness near the decision boundary. Exact split-conformal calibration on held-out data preserves distribution-free marginal coverage, which also upper-bounds the probability of an incorrect threshold decision that is not routed. Across various scientific and engineering datasets, TACR reduces simulator deferrals by 14–75% relative to standard conformal prediction at the same coverage target. Against a variant without threshold-local weighting but with similar predictor accuracy, TACR further reduces deferrals by 10–24% on four datasets. These results show that optimizing interval allocation for routing can reduce simulator calls without weakening the standard conformal guarantee.

1 INTRODUCTION

Engineering simulations, e.g., computational fluid dynamics (CFD), finite element analysis (FEA), and electronic design automation (EDA), are cornerstones of modern design workflows. The cost of high-fidelity simulation varies substantially with the physical problem, numerical method, resolution, hardware, and implementation. Reported scientific and engineering workflows commonly require minutes to days per simulation, which makes repeated simulator calls in a design loop a major computational bottleneck (Bonnet et al., 2022; Ashton et al., 2024; Ananthan et al., 2024; Jiang et al., 2024; Chai et al., 2023).

Recently, deep learning surrogates, including GNN-based MeshGraphNets (Pfaff et al., 2021; Nabian et al., 2024), Transformer-based neural fields (Adams et al., 2025; Wu et al., 2024; Luo et al., 2025; Alkin et al., 2025), and Neural Operators (Li et al., 2021; 2023; Ranade et al., 2025), have been developed to accelerate high-fidelity simulations. Once trained, these models can generally evaluate new inputs more cheaply than conventional simulators on the order of milliseconds to seconds. Avoiding even a fraction of simulator calls can yield substantial computational savings.

In scientific and engineering applications, particularly in safety-critical settings, realizing the computational savings from a neural surrogate requires knowing when the surrogate is sufficiently reliable for the downstream decision. Blindly relying on the surrogate's predictions risks missing engineering violations that the full simulator can identify. This motivates a per-input routing problem: use the surrogate when its prediction is sufficiently decisive for the downstream threshold test, and otherwise defer the input to the high-fidelity simulator. The two outcomes have asymmetric costs: simulation incurs substantial computation, whereas relying on an erroneous surrogate prediction can lead to an incorrect downstream decision.

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.

Contributions. Our contributions are as follows: (i) We formulate the surrogate-simulation handoff as threshold routing under conformal coverage and identify the flexibility of the nonconformity score as an opportunity to optimize interval allocation for routing rather than uncertainty estimation (§3.1); (ii) We propose Threshold-Aware Conformal Routing (TACR), whose threshold-aware σ-shaping objective concentrates interval tightness near the decision boundary through importance weighting (§3.2); (iii) We show that, across various scientific and engineering datasets, TACR reduces simulator deferrals by **14–75%** relative to standard conformal prediction at the same coverage target. Against a variant without threshold-local weighting but with similar predictor accuracy, TACR further reduces deferrals by **10–24%** on four datasets, showing an additional benefit from threshold-local weighting (§4.2); (iv) We further show that a deep-ensemble baseline targeting predictive uncertainty leads to more deferrals, providing empirical evidence that uncertainty estimation and routing-efficient interval allocation need not share the same objective (§4.2).

2 Related Work

Related work spans calibrated uncertainty for scientific surrogates, conformal prediction (CP), conformalized quantile regression (CQR), conformal efficiency optimization, decision-aware conformal methods, and selective prediction or model routing. Existing conformal methods provide coverage guarantees or optimize the generic set size. Decision-aware and deferral methods incorporate downstream utility or fallback decisions. To our knowledge, existing methods do not jointly learn to allocate conformal interval width according to distance from a fixed decision threshold while explicitly routing ambiguous inputs to a high-fidelity simulator. Our Threshold-Aware Conformal Routing (TACR) combines these elements: it retains split-conformal marginal coverage, optimizes interval allocation for the downstream routing decision, and invokes simulation when the calibrated interval intersects the threshold. Table 1 gives a compact comparison. (See Appendix A for details.)

Table 1: **Comparison of related approaches.** Representative methods cover subsets of the components combined by TACR. "Guarantee" denotes a distribution-free coverage or risk guarantee, whose target differs across rows; "CP efficiency" denotes direct optimization of conformal set size or interval width; and "abstain / route" indicates methods whose primary problem formulation is selective prediction, deferral, or model routing.

Approach	Guarantee	CP efficiency objective	Decision-aware	Threshold-local	Abstain / route
Standard CP / CQR[a]	✓	-	-	-	-
Operator-learning CP[b]	✓	-	-	-	-
Conformal efficiency optimization[c]	✓	✓	-	-	-
Learning to defer[d]	-	-	✓	-	✓
Conformalized selective regression[e]	✓	-	-	-	✓
Decision-aware CP[f]	✓	-	✓	-	-
Risk-controlled surrogate routing[g]	✓	-	✓	-	✓
Surrogate verification via CP[h]	✓	-	-	-	-
TACR (ours)	✓	✓	✓	✓	✓

[a] (Vovk et al., 2005; Romano et al., 2019); [b] (Ma et al., 2024); [c] (Stutz et al., 2022; Noorani et al., 2026; Lei & Bellotti, 2023; Xie et al., 2024; Su et al., 2026); [d] (Madras et al., 2018; Mozannar & Sontag, 2020); [e] (Sokol et al., 2026); [f] (Lekeufack et al., 2024; Cortes-Gomez et al., 2025; Yeh et al., 2025); [g] (Uddin et al., 2026); [h] (Qin et al., 2024). Standard CP, conformalized selective regression, and conformal-efficiency methods provide marginal coverage; operator-learning CP provides a finite-sample guarantee on average pointwise coverage across the output domain; decision-aware methods provide coverage or decision-risk control; risk-controlled routing bounds routed-set violations; and surrogate verification provides specification-level probabilistic validity. The coverage guarantee for conformalized selective regression is not conditional on acceptance.

3 Threshold-Aware Conformal Routing (TACR)

Figure 1 illustrates the key idea of TACR: conformal intervals determine whether to use the surrogate or route to simulation, while interval width is allocated relative to the decision threshold τ to reduce simulator calls. We next formalize the routing objective, training objective, and split-conformal calibration procedure.

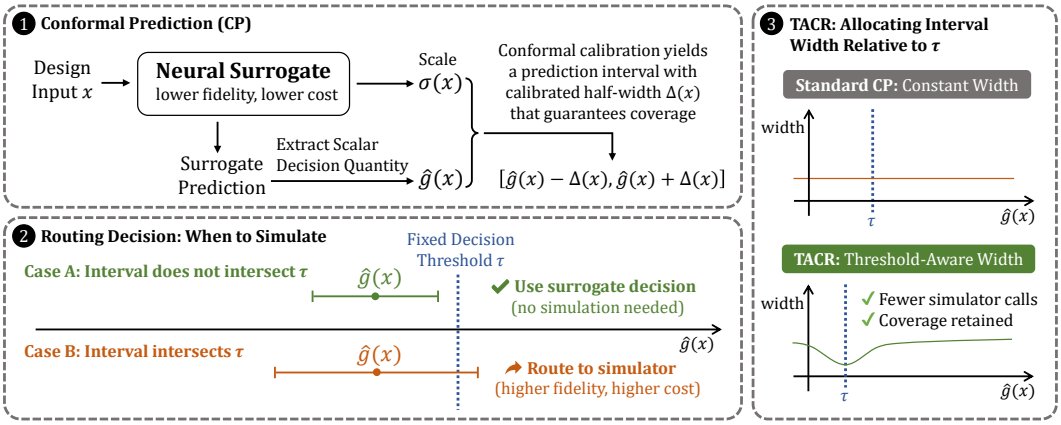

Figure 1: **Overview of Threshold-Aware Conformal Routing.** The calibrated interval determines whether to use the surrogate decision or route to simulation. TACR allocates interval width relative to τ to reduce simulator calls while retaining conformal coverage.

3.1 Problem Formulation and Routing Objective

A broad class of downstream engineering decisions can be expressed as threshold tests on scalar-valued functionals of high-dimensional fields produced by simulations. These functionals may take the form of integrals, extrema, percentiles, or other task-specific summaries of the full field. This enables decisions such as whether drag or peak stress exceeds a prescribed limit. We focus on this class of specification-driven decisions. Let x denote the simulator input, y the high-fidelity output, and $\hat{y} = f_\theta(x)$ the surrogate prediction. A functional $g(\cdot)$ maps each output to a scalar decision quantity, giving the true value $g(y)$ and surrogate estimate $\hat{g}(x) = g(f_\theta(x))$. Both g and the decision threshold τ are specified before training. For notational convenience, define the true and surrogate threshold decisions as $d(y) = \mathbb{1}[g(y) \le \tau]$ and $\hat{d}(x) = \mathbb{1}[\hat{g}(x) \le \tau]$, respectively.

We augment the surrogate with a positive, sample-dependent scale $\sigma(x) > 0$. After split-conformal calibration, this scale defines the interval

$$C(x) = [\hat{g}(x) - \hat{q}\sigma(x),\ \hat{g}(x) + \hat{q}\sigma(x)], \tag{1}$$

where $\hat{q}$ is computed on held-out calibration data. If $C(x)$ lies entirely above or below τ, we use the surrogate's threshold decision; if it crosses τ, we route the sample to simulation. Equivalently,

$$r(x) = \mathbb{1}[|\hat{g}(x) - \tau| \le \hat{q}\sigma(x)]\,, \tag{2}$$

where $r(x) = 1$ denotes routing to simulation.

The routing rule links conformal coverage to decision safety. Whenever $r(x) = 0$ and $g(y) \in C(x)$, the entire interval and the true decision quantity lie on the same side of τ, so $\hat{d}(x) = d(y)$. Consequently, a wrong decision that is not routed can occur only when the conformal interval fails to cover $g(y)$ (see Section 3.3 for formalization of the resulting risk bound).

Our goal is to reduce the expected routing rate while retaining split-conformal coverage at a prescribed level $1 - \alpha$:

$$\min_{\theta,\phi}\ \mathbb{E}[r(X)] \qquad \text{subject to} \qquad \Pr(g(Y) \in C(X)) \ge 1 - \alpha, \tag{3}$$

where ϕ parameterizes σ (§3.2). The coverage constraint is enforced after training by standard split-conformal calibration, rather than by treating α as a direct budget on decision errors. Directly optimizing the routing rate is difficult because the routing indicator in Eq. 2 is discrete and $\hat{q}$ is determined only after training. We use a differentiable training objective designed to reduce routing. The key observation is that routing depends on the average interval width and where that width is allocated, i.e., width near τ can trigger simulation, whereas samples farther from τ can tolerate wider intervals without changing the decision. This motivates TACR to concentrate interval tightness near the decision boundary rather than requiring σ to represent predictive uncertainty.

3.2 THRESHOLD-AWARE TRAINING OBJECTIVE

Objective overview. Let $\mathcal{B}$ denote a training mini-batch. For each $i \in \mathcal{B}$, we define

$$\hat{g}_i = g(f_\theta(x_i)), \qquad \sigma_i = \sigma(x_i). \tag{4}$$

The complete training objective and its two scale-learning terms are

$$\mathcal{L} = \mathcal{L}_{\text{pred}} + \lambda \left(\mathcal{L}_{\text{route}} + \mu \mathcal{L}_{\text{scale}} \right), \tag{5}$$

$$\mathcal{L}_{\text{route}} = \tilde{q}_{\mathcal{B}} \frac{1}{|\mathcal{B}|} \sum_{i \in \mathcal{B}} (1 + \beta \text{prox}_i)\, \sigma_i, \tag{6}$$

$$\mathcal{L}_{\text{scale}} = \frac{1}{|\mathcal{B}|} \sum_{i \in \mathcal{B}} \left(\sigma_i + \frac{1}{\sigma_i} \right). \tag{7}$$

Here, $\mathcal{L}_{\text{pred}}$ denotes the surrogate's prediction objective, which may vary across architectures and datasets. The quantity $\tilde{q}_{\mathcal{B}}$ approximates the conformal multiplier during training, and prox_i measures the proximity of sample i to the decision threshold. The coefficient $\lambda \geq 0$ weights the two scale-learning terms relative to prediction, $\mu \geq 0$ controls scale regularization, and $\beta \geq 0$ controls the additional emphasis near the threshold. We define $\tilde{q}_{\mathcal{B}}$ and prox_i next.

Differentiable conformal width. For each sample, define the normalized score

$$s_i = |g(y_i) - \hat{g}_i| / \sigma_i. \tag{8}$$

The final conformal multiplier $\hat{q}$ is unavailable during training because it is computed later on held-out calibration data. We construct a differentiable mini-batch approximation from the normalized scores $\{s_i\}_{i \in \mathcal{B}}$:

$$\tilde{q}_{\mathcal{B}} = \text{SoftQuantile}_{1-\alpha}(\{s_i\}_{i \in \mathcal{B}}). \tag{9}$$

This approximation is computed separately for each training mini-batch and is used only for optimization; implementation details are given in Appendix E. Including $\tilde{q}_{\mathcal{B}}$ couples the learned scales to the calibration mechanism, making $\tilde{q}_{\mathcal{B}} \sigma_i$ a training-time approximation to the calibrated half-width. The objective can shape the relative allocation of calibrated width rather than shrink raw scales without accounting for recalibration.

Threshold-local weighting. We define

$$\text{prox}_i = \exp\left(-\frac{(\text{stopgrad}(\hat{g}_i) - \tau)^2}{2\,\text{bw}^2} \right), \tag{10}$$

where $\text{bw} > 0$ controls the neighborhood around the threshold and is fixed during training (see Section 4.1 for the setting used in our experiments). The weight approaches 1 near τ and decays toward 0 with distance. With $\beta \geq 0$, the factor $1 + \beta \text{prox}_i$ applies baseline width pressure to every sample and additional pressure near the decision boundary, where calibrated width is most likely to change the routing decision.

Stop-gradient in the proximity weight. The dependence of prox_i on $\hat{g}_i$ can create a shortcut when differentiated end to end. Because prox_i decreases as $|\hat{g}_i - \tau|$ grows, the predictor can reduce the additional weight assigned to a sample by pushing its decision value away from the threshold, even when doing so does not improve prediction accuracy. We stop gradients through $\hat{g}_i$ when computing prox_i, which blocks this degenerate shortcut.

Objective interpretation. Routing occurs when the calibrated half-width $\hat{q}\sigma_i$ equals or exceeds the predicted threshold margin $|\hat{g}_i - \tau|$. Accordingly, $\mathcal{L}_{\text{route}}$ minimizes a threshold-weighted average of the training-time calibrated half-widths $\tilde{q}_{\mathcal{B}} \sigma_i$ as a differentiable proxy for routing efficiency. Ignoring the numerical stabilizer used in implementation, a uniform rescaling $\sigma_i \mapsto c\sigma_i$ changes the normalized-score quantile as $\tilde{q}_{\mathcal{B}} \mapsto \tilde{q}_{\mathcal{B}}/c$, leaving this calibrated-width term unchanged. The scale regularizer breaks the resulting global scale indeterminacy and discourages extreme values of σ_i.

Scale-head instantiation. We parameterize the positive scale as $\sigma(x) = \sigma_\phi(h_\theta(x))$, where $h_\theta(x)$ is the surrogate's penultimate backbone representation and ϕ denotes the scale-head parameters. The lightweight head reuses backbone features and outputs one positive scale per sample. See Appendix E for details on the architectural specifications.

Training schedule. The scale head and threshold-aware objective build on the predictor's outputs and backbone features. We train in two stages: first optimizing the predictor with $\mathcal{L}_{\text{pred}}$, then activating the scale head and jointly optimizing Eq. 5. We use this schedule for all datasets, including CircuitNet, whose ResNet18 encoder is initialized from ImageNet weights. See Appendix E for the stage lengths and optimization settings.

3.3 Calibration and Deployment

After training, we freeze the predictor and scale function and perform standard split-conformal calibration on a held-out set indexed by $\mathcal{I}_{\text{cal}}$, with $n_{\text{cal}} = |\mathcal{I}_{\text{cal}}|$. For every calibration sample, we compute

$$s_i = |g(y_i) - \hat{g}(x_i)|/\sigma(x_i), \qquad i \in \mathcal{I}_{\text{cal}}. \tag{11}$$

Let $s_{(1)} \leq \cdots \leq s_{(n_{\text{cal}})}$ denote the sorted calibration scores. The finite-sample corrected conformal multiplier is

$$k = \lceil (n_{\text{cal}} + 1)(1 - \alpha) \rceil, \qquad \hat{q} = s_{(k)}, \tag{12}$$

with the standard convention $\hat{q} = \infty$ if $k > n_{\text{cal}}$. At deployment, we form $C(x)$ using Eq. 1 and apply the routing rule in Eq. 2.

Under the standard exchangeability assumptions of split conformal prediction, the resulting interval satisfies distribution-free marginal coverage,

$$\Pr(g(Y) \in C(X)) \geq 1 - \alpha. \tag{13}$$

Since a wrong threshold decision that is not routed implies noncoverage,

$$\begin{aligned} \Pr\Big(\hat{d}(X) \neq d(Y),\, r(X) = 0\Big) &= \Pr\Big(\hat{d}(X) \neq d(Y),\, r(X) = 0,\, g(Y) \notin C(X)\Big) \\ &\leq \Pr(g(Y) \notin C(X)) \leq \alpha. \end{aligned} \tag{14}$$

The first inequality can be strict: a value outside the interval may still lie on the same side of τ as the surrogate decision. Thus, α specifies an interval-coverage level and provides a conservative upper bound on the unconditional miss probability; it is not a miss budget that the method is expected to exhaust. Importantly, the guarantee does not require σ itself to estimate predictive uncertainty.

4 Empirical Evaluation

We first describe the datasets, baselines, metrics, and evaluation protocol. We then compare routing performance and study sensitivity to the decision threshold, backbone, and threshold-local weighting strength.

4.1 Experimental Design

Datasets. We evaluate TACR on six datasets spanning EDA, fluid mechanics, and structural mechanics, with convolutional, graph, transformer, and MLP surrogates: CircuitNet (Jiang et al., 2024), Darcy Flow (Takamoto et al., 2022), AirfRANS (Bonnet et al., 2022), AhmedML (Ashton et al., 2024), ShapeNet-Car (Umetani & Bickel, 2018), and a synthetic parametric-beam dataset (Table 2). For field-valued tasks, g extracts the scalar decision quantity from the predicted field; for Beams, the surrogate predicts log-transformed peak stress directly. This diversity tests whether TACR generalizes across backbones and output representations.

Baselines. Within each dataset, all methods use the same predictor backbone, with method-specific training objectives and output heads where required: **Standard conformal prediction (Standard CP)** uses a constant scale, σ=1; **Conformalized Quantile Regression (CQR)** (Romano et al., 2019) uses conditional-quantile heads with $\sigma = (q_{hi} - q_{lo})/2$; **Threshold-Weighted Uncertainty Quantification (TW-UQ)** adds a threshold-weighted MSE on the scalar decision quantity and trains a scale head with Gaussian negative log likelihood; **Uniform** is identical to TACR except that it sets β=0, removing threshold-local weighting; **Deep Ensemble** (Lakshminarayanan et al., 2017) uses the predictive standard deviation of five predictor members as σ; and our **TACR** uses Eq. 5 with β=16 by default. Together, these baselines distinguish the effects of learned input-dependent

Table 2: **Datasets and decision tasks.** N is the number of samples; p95 denotes the 95th percentile. IC, EDA, and CFD denote integrated circuit, electronic design automation, and computational fluid dynamics; C_L and C_D are lift and drag coefficients.

Dataset	Physics	N	Input $\rightarrow$ target y	Decision $g(y)$
CircuitNet	IC congestion (EDA)	3,456	feature map (3 channels) $\rightarrow$ congestion map	p95(y)
Darcy Flow	2D steady-state flow	10,000	coefficient field $128^2 \rightarrow$ pressure field u	p95($\lvert u \rvert$)
AirfRANS	2D airfoil CFD	1,000	point cloud (7 features) $\rightarrow$ surface pressure	C_L
AhmedML	3D bluff-body CFD	498	point cloud (6 features) $\rightarrow$ surface pressure	C_D
ShapeNet-Car	3D vehicle CFD	888	point cloud (xyz+normals) $\rightarrow$ surface pressure	C_D
Beams	parametric beam (synthetic)	2,000	design vector $\rightarrow \log_{10}$ peak stress	$g(y) = y$

scaling, threshold-local weighting beyond the Uniform objective, and uncertainty-oriented scaling for routing efficiency. See Appendix E for the full optimization settings.

Metrics. We report the simulation rate (SIM%), the percentage of test inputs routed to simulation, corresponding to the event $r(X) = 1$; the miss rate (MISS%), the percentage for which the surrogate decision is incorrect and the input is not routed, corresponding to $\hat{d}(X) \neq d(Y), r(X) = 0$; and g-MAE, the test-set mean of $|\hat{g}(x) - g(y)|$, which measures error in the scalar decision quantity. SIM% is our primary efficiency metric because it directly measures the fraction of inputs that incur a high-fidelity simulator call and serves as a direct proxy for simulation cost, while MISS% quantifies the accompanying decision risk. For additional routing diagnostics, we define NEEDS% as the method-specific percentage of test inputs satisfying $\hat{d}(X) \neq d(Y)$. See Appendix C for metric definitions and for routing precision, recall, and F1 metrics, where we treat $\hat{d}(X) \neq d(Y)$ as the positive class.

Training and calibration. The data splits are fixed before model training, with dataset-specific proportions reported in Appendix E. The calibration set is used only to compute the final conformal multiplier, while the training-time interpolated quantile is computed exclusively from training mini-batches. We set $\alpha = 0.1$, corresponding to 90% target coverage, and use the 70th percentile of the training-set decision quantity as the default threshold unless otherwise noted. To avoid dataset-specific tuning, we use the same global heuristic settings for the routing-loss coefficients and threshold bandwidth across all datasets and seeds. For TACR, all datasets use predictor-only training followed by joint optimization of the predictor and scale head. See Appendix E for the exact split sizes, baseline objectives, hyperparameters, and optimization settings.

Reporting protocol. We evaluate each non-ensemble method over five random seeds, and report the mean and standard deviation. The diagnostic controls use the seed counts stated in their captions. Deep Ensemble uses five independently initialized predictors per dataset and is substantially more expensive than the single-model methods because all five predictors must be trained and evaluated (Mouli et al., 2024). We report point estimates rather than seed variation since the results comprise one five-member ensemble run per dataset.

4.2 MAIN RESULTS

Table 3 shows the results at the common nominal coverage level $1 - \alpha = 90\%$ using a fixed decision threshold for each dataset. We prioritize SIM% because every routed input incurs a high-fidelity simulator call, which makes it a direct proxy for simulation cost. TACR lowers the mean SIM% relative to Standard CP on all six datasets and attains the lowest mean on four. MISS% provides the corresponding decision-risk measure and is theoretically upper-bounded by the noncoverage rate. For TACR, it remains below 1% on five datasets and is 2.38% on AhmedML, which is well below the nominal 10% noncoverage level. The g-MAE diagnostic measures the accuracy of the scalar decision quantity, which can affect routing efficiency by changing how accurately the surrogate resolves the threshold decision. TACR has the lowest or second-lowest mean g-MAE on five of six datasets, with Beams as the exception. We distinguish below between the overall gains relative to Standard CP and the contribution of threshold-local interval allocation. Section 4.4 compares the methods across multiple values of α.

Threshold-local weighting contributes to the routing gains. Table 3 provides our primary evidence that threshold-local weighting contributes beyond changes in predictor accuracy. Relative to Uniform, TACR reduces SIM% by 10–24% on CircuitNet, Darcy, AirfRANS, and Beams. The differences on AhmedML and ShapeNet-Car are smaller than the reported standard deviations for

Table 3: **Main routing results across six scientific and engineering datasets.** SIM%, the fraction of inputs sent to high-fidelity simulation, is our primary cost metric; MISS% measures incorrect surrogate decisions that escape simulation. At the common 90% coverage target, TACR reduces SIM% relative to Standard CP on all six datasets, attains the lowest mean SIM% on four, and keeps MISS% below 2.5% throughout. The decision-quantity mean absolute error, g-MAE, is an auxiliary diagnostic of predictor accuracy. Entries are mean±std over five seeds; the ensemble uses one five-member run. Lower is better. Bold and underline mark the lowest and second-lowest mean within each dataset and metric, respectively; exact ties share the same formatting. On Darcy, † marks CQR and TACR, whose mean SIM% values differ by less than either reported standard deviation.

	Method	CircuitNet	Darcy	AirfRANS	AhmedML	ShapeNet-Car	Beams
SIM%	Standard CP	46.4±0.6	14.8±1.1	19.8±2.0	31.7±5.1	46.4±3.2	9.4±0.7
	CQR	65.9±1.5	12.5±5.6†	19.1±3.3	27.9±2.4	38.3±3.6	7.8±0.6
	Uniform (β=0)	15.3±0.8	14.1±0.7	9.7±0.9	**25.5±4.7**	36.4±3.7	7.5±0.6
	TW-UQ	22.5±1.0	13.6±1.2	10.5±1.0	36.6±1.8	48.2±4.8	7.2±0.2
	Deep Ensemble	50.5	**8.3**	28.0	29.7	48.6	14.2
	TACR	**11.6±0.5**	12.7±0.4†	**7.4±0.9**	25.7±4.7	**34.7±4.2**	**5.7±0.2**
MISS%	Standard CP	0.37±0.05	0.03±0.02	0.20±0.24	1.98±0.63	0.78±0.27	**0.20±0.10**
	CQR	**0.09±0.09**	0.11±0.12	0.30±0.40	**1.78±1.15**	0.67±0.22	**0.20±0.19**
	Uniform (β=0)	0.23±0.07	0.02±0.02	0.10±0.20	2.57±0.49	0.56±0.35	0.35±0.12
	TW-UQ	0.12±0.00	0.13±0.05	0.40±0.20	**1.78±0.97**	**0.45±0.42**	0.40±0.12
	Deep Ensemble	0.12	**0.00**	**0.00**	2.97	0.56	0.25
	TACR	0.44±0.11	0.06±0.05	0.40±0.20	2.38±0.49	0.56±0.35	0.60±0.30
g-MAE	Standard CP	0.0082±0.0002	0.0135±0.0005	0.0777±0.0063	0.0172±0.0011	0.0132±0.0006	0.1314±0.0023
	CQR	0.0127±0.0005	0.0131±0.0004	0.0678±0.0141	0.0153±0.0017	0.0110±0.0008	0.1302±0.0026
	Uniform (β=0)	**0.0023±0.0001**	0.0129±0.0003	**0.0368±0.0026**	0.0149±0.0008	0.0101±0.0009	0.1320±0.0025
	TW-UQ	0.0053±0.0002	0.0152±0.0005	0.0485±0.0010	0.0208±0.0032	0.0140±0.0006	0.1329±0.0017
	Deep Ensemble	0.0083	**0.0127**	0.0694	0.0190	0.0129	**0.1286**
	TACR	0.0024±0.0001	0.0129±0.0003	0.0369±0.0032	**0.0148±0.0007**	**0.0095±0.0008**	0.1336±0.0024

both methods. Uniform retains TACR's decision-level training signal but removes threshold-local weighting, while attaining similar g-MAE and NEEDS% (see Table 9 in Appendix C). Because the two methods have similar predictor accuracy, their routing difference on the first four datasets is unlikely to result from predictor quality alone. In contrast, the comparison with Standard CP reflects both changes in predictor accuracy and interval allocation. An additional control in Appendix B shows that adding explicit g-level supervision to the baselines does not reproduce the routing gains on CircuitNet and AirfRANS. Together, these comparisons identify threshold-local interval allocation as a contributor to the routing gains.

Predictive-uncertainty scales do not reduce simulator use. CQR, TW-UQ, and Deep Ensemble adapt interval width using conditional quantiles, a likelihood-based scale, and ensemble dispersion, respectively. These constructions are designed to reflect predictive uncertainty, but they do not explicitly optimize interval allocation around the decision threshold τ. We see in Table 3 that at the common coverage target, TW-UQ routes more inputs than TACR on all six datasets, and CQR and Deep Ensemble do so on five of the six. Uncertainty-aware scaling need not reduce SIM%.

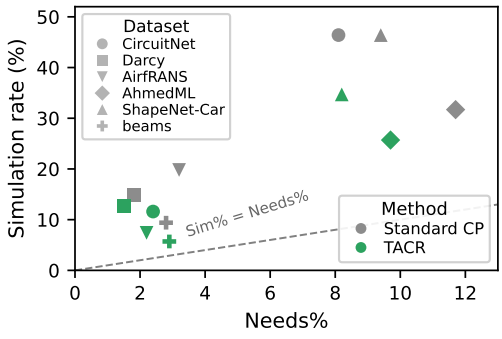

Figure 2: **Routing relative to the zero-miss reference.** Compared with Standard CP, TACR requires less excess routing on all six datasets. The dashed line, SIM% = NEEDS%, marks the minimum routing rate under zero-miss oracle routing for each fixed surrogate.

TACR has less excess routing above the zero-miss bound. For a fixed surrogate, the oracle zero-miss router routes exactly those inputs on which the surrogate makes an incorrect threshold decision. This yields SIM% = NEEDS%, which is the minimum simulation rate achievable with zero misses. Figure 2 shows that TACR has less excess routing above this bound than Standard CP on all six datasets; Figure 4 in Appendix C includes all evaluated methods. Since the evaluated methods allow nonzero MISS%, this bound serves as a diagnostic reference rather than an equal-safety operating point.

Table 4: **Sensitivity to the decision threshold.** Threshold-local weighting remains effective as the threshold changes: TACR attains the lowest mean SIM% in seven of eight settings and ties Uniform in the remaining setting. Thresholds are percentiles of the training-set decision quantity. Reported results are mean±std over five seeds, and the main experiments use p70.

CircuitNet

Method	p50	p70	p90	p95
Standard CP	47.5±0.8	46.4±0.6	16.4±0.4	5.3±0.5
CQR	69.4±5.0	65.9±1.5	31.1±4.5	12.7±1.7
Uniform	24.1±0.7	15.3±0.8	6.4±0.6	**2.6±0.3**
TW-UQ	25.1±1.4	22.5±1.0	10.9±0.5	6.0±0.2
TACR	**19.7±0.9**	**11.6±0.5**	**5.2±0.4**	**2.6±0.3**

AirfRANS

Method	p50	p70	p90	p95
Standard CP	18.1±1.4	19.8±2.0	15.2±0.4	9.4±1.5
CQR	11.7±1.9	19.1±3.3	13.7±1.4	7.5±0.7
Uniform	7.4±0.6	9.7±0.9	8.2±1.3	6.1±0.2
TW-UQ	6.5±0.5	10.5±1.0	10.5±0.8	6.1±0.7
TACR	**3.7±0.2**	**7.4±0.9**	**4.5±0.4**	**3.1±1.2**

Table 5: **Sensitivity to threshold-local weighting.** Stronger weighting monotonically reduces mean SIM% on CircuitNet from 13.4% to 9.7%, while MISS% shows no monotonic increase. Results are mean±std over five seeds; the main experiments use $\beta = 16$, and bold marks the lowest mean in each metric.

β	SIM%	MISS%
4	13.4±0.7	**0.28±0.17**
8	12.5±0.3	0.37±0.15
16	11.6±0.5	0.44±0.11
32	11.2±0.4	0.39±0.16
64	**9.7±0.8**	0.39±0.17

Table 6: **Sensitivity to the surrogate backbone.** TACR retains the lowest mean SIM% when the AirfRANS backbone changes from GraphSAGE to a four-head Graph Transformer, providing evidence of transfer across these architectures. SIM% is mean±std over five seeds, and MISS% is a percentage.

Backbone	Method	SIM%	MISS%
GraphSAGE	Standard CP	19.8±2.0	0.20
	CQR	19.1±3.3	0.30
	Uniform (β=0)	9.7±0.9	0.10
	TACR	**7.4±0.9**	0.40
Graph-Transformer	Standard CP	18.3±1.7	0.10
	CQR	15.2±0.9	0.10
	Uniform (β=0)	8.6±1.4	0.10
	TACR	**6.3±0.5**	0.20

4.3 ROBUSTNESS AND SENSITIVITY ANALYSES

We examine whether the routing gains persist across changes in the decision threshold, threshold-local weighting strength, and surrogate backbone.

TACR remains effective across decision thresholds. The threshold τ is an application specification fixed before deployment. Table 4 varies it over the 50th, 70th, 90th, and 95th percentiles of the training-set decision quantity on CircuitNet and AirfRANS. TACR attains the lowest mean SIM% in seven of eight settings and ties Uniform in the remaining one, showing that its routing gains are not specific to the p70 threshold used in the main experiments.

Sensitivity to threshold-local weighting. Table 5 varies β on CircuitNet while holding all other settings fixed. Mean SIM% decreases monotonically from 13.4% at $\beta = 4$ to 9.7% at $\beta = 64$. MISS% remains between 0.28% and 0.44% without a monotonic increase. This dose-response pattern provides additional evidence that stronger threshold-local weighting systematically improves routing efficiency, rather than the gain at the default $\beta = 16$ arising from a single favorable operating point. We retain the prespecified $\beta = 16$ used throughout the main experiments. The sweep is intended to characterize sensitivity rather than to tune β for individual datasets.

Routing gains are consistent across two surrogate backbones. Table 6 compares GraphSAGE with a four-head Graph Transformer on AirfRANS. The method ordering is preserved, with TACR attaining the lowest mean SIM% under both backbones. This provides evidence that the observed routing gains are not specific to the GraphSAGE backbone used in the main AirfRANS experiments.

4.4 POST-HOC COST-RISK TRADE-OFFS

The main results use models trained with $\alpha = 0.1$. We next examine whether a model trained at this single coverage level can still support useful deployment-time trade-offs through recalibration alone. Without retraining, we recompute the exact conformal quantile $\hat{q}(\alpha)$ over $\alpha \in [0.02, 0.30]$ and evaluate the resulting SIM%–MISS% operating points. Figure 3 shows that changing the calibration level produces substantial variation in simulator use and decision risk. This indicates that a fixed

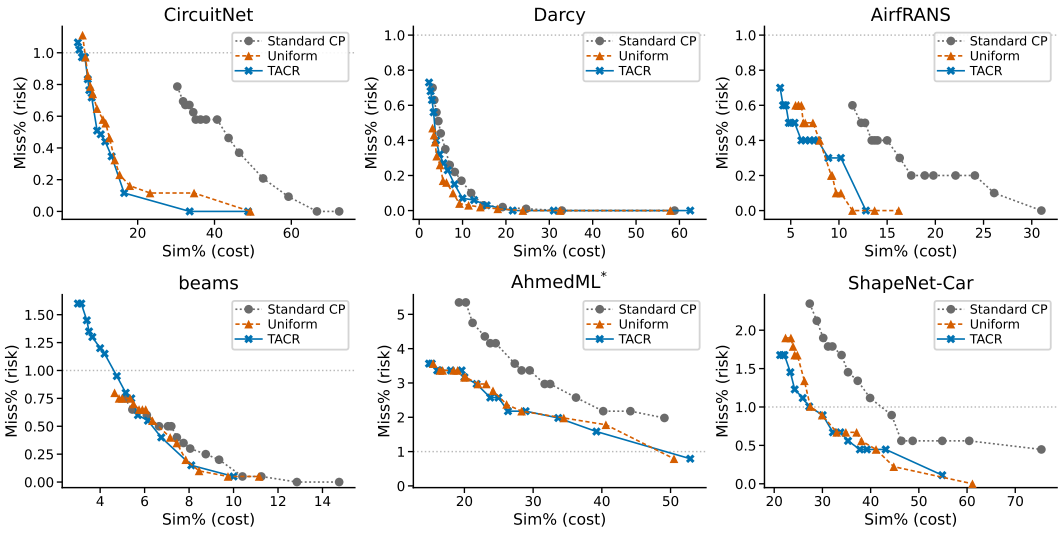

Figure 3: **Post-hoc cost–risk trade-offs.** Models are trained once with $\alpha = 0.1$ and then recalibrated over $\alpha \in [0.02, 0.30]$ by recomputing only the exact conformal quantile, without retraining. Trade-offs vary by dataset and operating point: learned-scale methods often require fewer simulator calls than Standard CP at comparable MISS%, while crossings between TACR and Uniform show that neither uniformly dominates across all recalibrated operating points. Curves aggregate five seeds; the lower left is preferred. The dotted horizontal line marks a 1% miss rate. For AhmedML, Standard CP and Uniform use a learning rate of 10^{-3} rather than the 5×10^{-4} used in the main table.

trained model can be adapted post hoc to different deployment requirements. Smaller α generally produces wider intervals and more routing, while empirical MISS% may decrease. Since conformal ranks and test-set counts are discrete, the curves are stepwise and empirical MISS% need not vary monotonically. The relative ordering is dataset- and operating-point-dependent, with crossings between TACR and Uniform in some regimes. Since α also enters the training-time soft quantile, these curves characterize the post-hoc flexibility of models trained at $\alpha = 0.1$, rather than separately optimized performance at each coverage level.

5 CONCLUSION

Learned surrogates can reduce the cost of high-fidelity simulation, but their errors can alter downstream decisions. We formulate the surrogate-simulator handoff for fixed scalar thresholds as threshold-aware routing with split-conformal coverage. Exploiting conformal validity for any nonconformity score fixed before calibration, our proposed TACR method learns to allocate interval width according to its effect on routing. Across various scientific and engineering datasets, TACR reduces the fraction of inputs routed to simulation by 14–75% relative to standard conformal prediction at the same nominal coverage target. Against a variant without threshold-local weighting but with similar predictor accuracy, TACR further reduces deferrals by 10–24% on four datasets. These results show that routing-aware interval allocation can reduce simulator calls without weakening the standard conformal guarantee.

Limitations and future work. TACR currently addresses decisions defined by a fixed scalar threshold. Extending it to multiple simultaneous specifications would require a joint routing rule and an appropriate calibration target. In addition, its guarantee is marginal rather than conditional, so the stratified results in Appendix D are empirical diagnostics and do not establish local risk control. Routing efficiency depends on surrogate accuracy and on the distribution of decision quantities around the threshold, and may vary across tasks. Extensions to group-conditional calibration, multi-constraint decisions, and online updating with newly simulated samples are promising future directions.

Reproducibility statement

Sections 3 and 4.1 specify the method, calibration and routing procedures, datasets, baselines, metrics, and evaluation protocol; Section 3.3 states the assumptions underlying the finite-sample guarantee. Appendix E provides the exact data splits and preprocessing, dataset-specific architectures and scale heads, prediction objectives, hyperparameters, optimization schedules, and calibration details. Dataset and backbone sources are cited in Section 4.1 and Table 2, and the captions identify the seed aggregation and evaluation settings used for the reported results.

AI use statement

In this work, we used generative AI tools to retrieve and organize relevant literature; assist with code development, code auditing, result aggregation, and figure preparation; and polish portions of the manuscript. The authors reviewed all AI-assisted text, checked cited literature against the original sources, and inspected and tested AI-assisted code and numerical outputs. Generative AI was not used to generate synthetic datasets or prove mathematical claims. We take responsibility for the final content of this work, including text, claims, code, and artifacts produced with the aid of generative AI.

References

Corey Adams, Rishikesh Ranade, Ram Cherukuri, and Sanjay Choudhry. Geotransolver: Learning physics on irregular domains using multi-scale geometry aware physics attention transformer. *arXiv preprint arXiv:2512.20399*, 2025.

Benedikt Alkin, Maurits Bleeker, Richard Kurle, Tobias Kronlachner, Reinhard Sonnleitner, Matthias Dorfer, and Johannes Brandstetter. AB-UPT: Scaling neural CFD surrogates for high-fidelity automotive aerodynamics simulations via anchored- branched universal physics transformers. *Transactions on Machine Learning Research*, 2025. ISSN 2835-8856.

Alexander Amini, Wilko Schwarting, Ava Soleimany, and Daniela Rus. Deep evidential regression. *Advances in neural information processing systems*, 33:14927–14937, 2020.

Vidyasagar Ananthan, Neil Ashton, Nate Chadwick, Mariano Lizarraga, Danielle Maddix, Satheesh Maheswaran, Pablo Hermoso Moreno, Parisa M. Shabestari, Sandeep Sovani, Shreyas Subramanian, et al. Machine learning for road vehicle aerodynamics. In *WCX SAE World Congress Experience 288443*, 2024.

Anastasios N Angelopoulos and Stephen Bates. Conformal prediction: A gentle introduction. *Foundations and Trends in Machine Learning*, 16(4):494–591, 2023.

Neil Ashton, Danielle C. Maddix, Samuel Gundry, and Parisa M Shabestari. AhmedML: High-fidelity computational fluid dynamics dataset for incompressible, low-speed bluff body aerodynamics. *arXiv preprint arXiv:2407.20801*, 2024.

Florent Bonnet, Jocelyn Mazari, Paola Cinnella, and Patrick Gallinari. AirfRANS: High fidelity computational fluid dynamics dataset for approximating reynolds-averaged navier–stokes solutions. *Advances in Neural Information Processing Systems*, 35:23463–23478, 2022.

Zhuomin Chai, Yuxiang Zhao, Wei Liu, Yibo Lin, Runsheng Wang, and Ru Huang. CircuitNet: An open-source dataset for machine learning in vlsi cad applications with improved domain-specific evaluation metric and learning strategies. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 42(12):5034–5047, 2023.

Santiago Cortes-Gomez, Carlos Patino, Yewon Byun, Steven Wu, Eric Horvitz, and Bryan Wilder. Utility-directed conformal prediction: A decision-aware framework for actionable uncertainty quantification. In *International Conference on Learning Representations*, volume 2025, pp. 4227–4246, 2025.

Yarin Gal and Zoubin Ghahramani. Dropout as a bayesian approximation: Representing model uncertainty in deep learning. In *international conference on machine learning*, pp. 1050–1059. PMLR, 2016.

Xun Jiang, Yuxiang Zhao, Yibo Lin, Runsheng Wang, Ru Huang, et al. Circuitnet 2.0: An advanced dataset for promoting machine learning innovations in realistic chip design environment. In *The Twelfth International Conference on Learning Representations*, 2024.

Balaji Lakshminarayanan, Alexander Pritzel, and Charles Blundell. Simple and scalable predictive uncertainty estimation using deep ensembles. *Advances in neural information processing systems*, 30, 2017.

Haocheng Lei and Anthony Bellotti. Reliable prediction intervals with directly optimized inductive conformal regression for deep learning. *Neural Networks*, 168:194–205, 2023.

Jordan Lekeufack, Anastasios N Angelopoulos, Andrea Bajcsy, Michael I Jordan, and Jitendra Malik. Conformal decision theory: Safe autonomous decisions from imperfect predictions. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 11668–11675. IEEE, 2024.

Zongyi Li, Nikola Borislavov Kovachki, Kamyar Azizzadenesheli, Burigede Liu, Kaushik Bhattacharya, Andrew Stuart, and Anima Anandkumar. Fourier neural operator for parametric partial differential equations. In *International Conference on Learning Representations*, 2021.

Zongyi Li, Nikola Kovachki, Chris Choy, Boyi Li, Jean Kossaifi, Shourya Otta, Mohammad Amin Nabian, Maximilian Stadler, Christian Hundt, Kamyar Azizzadenesheli, et al. Geometry-informed neural operator for large-scale 3d pdes. *Advances in Neural Information Processing Systems*, 36:35836–35854, 2023.

Huakun Luo, Haixu Wu, Hang Zhou, Lanxiang Xing, Yichen Di, Jianmin Wang, and Mingsheng Long. Transolver++: An accurate neural solver for PDEs on million-scale geometries. In Aarti Singh, Maryam Fazel, Daniel Hsu, Simon Lacoste-Julien, Felix Berkenkamp, Tegan Maharaj, Kiri Wagstaff, and Jerry Zhu (eds.), *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 41432–41449. PMLR, 13–19 Jul 2025.

Ziqi Ma, David Pitt, Kamyar Azizzadenesheli, and Anima Anandkumar. Calibrated uncertainty quantification for operator learning via conformal prediction. *Transactions on Machine Learning Research*, 2024. ISSN 2835-8856.

David Madras, Toni Pitassi, and Richard Zemel. Predict responsibly: improving fairness and accuracy by learning to defer. *Advances in neural information processing systems*, 31, 2018.

Emilia Magnani, Nicholas Krämer, Runa Eschenhagen, Lorenzo Rosasco, and Philipp Hennig. Approximate bayesian neural operators: Uncertainty quantification for parametric PDEs. *Transactions on Machine Learning Research*, 2025. ISSN 2835-8856.

S Chandra Mouli, Danielle C. Maddix, Shima Alizadeh, Gaurav Gupta, Andrew Stuart, Michael W. Mahoney, and Bernie Wang. Using uncertainty quantification to characterize and improve out-of-domain learning for PDEs. In Ruslan Salakhutdinov, Zico Kolter, Katherine Heller, Adrian Weller, Nuria Oliver, Jonathan Scarlett, and Felix Berkenkamp (eds.), *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 36372–36418. PMLR, 21–27 Jul 2024.

Hussein Mozannar and David Sontag. Consistent estimators for learning to defer to an expert. In *International conference on machine learning*, pp. 7076–7087. PMLR, 2020.

Mohammad Amin Nabian, Chang Liu, Rishikesh Ranade, and Sanjay Choudhry. X-meshgraphnet: Scalable multi-scale graph neural networks for physics simulation. *arXiv preprint arXiv:2411.17164*, 2024.

Sima Noorani, Orlando Romero, Nicolo Dal Fabbro, Hamed Hassani, and George J. Pappas. Conformal risk minimization with variance reduction. In Emilija Perković and Daniel Malinsky (eds.), *Proceedings of the 42nd Conference on Uncertainty in Artificial Intelligence*, volume 337 of *Proceedings of Machine Learning Research*, pp. 4972–5002. PMLR, 17–21 Aug 2026.

Harris Papadopoulos, Kostas Proedrou, Volodya Vovk, and Alex Gammerman. Inductive confidence machines for regression. In *European conference on machine learning*, pp. 345–356. Springer, 2002.

Tobias Pfaff, Meire Fortunato, Alvaro Sanchez-Gonzalez, and Peter Battaglia. Learning mesh-based simulation with graph networks. In *International Conference on Learning Representations*, 2021.

Xin Qin, Yuan Xia, Aditya Zutshi, Chuchu Fan, and Jyotirmoy V Deshmukh. Statistical verification using surrogate models and conformal inference and a comparison with risk-aware verification. *ACM Transactions on Cyber-Physical Systems*, 8(2):1–25, 2024.

Rishikesh Ranade, Mohammad Amin Nabian, Kaustubh Tangsali, Alexey Kamenev, Oliver Hennigh, Ram Cherukuri, and Sanjay Choudhry. DoMINO: A decomposable multi-scale iterative neural operator for modeling large scale engineering simulations. *arXiv preprint arXiv:2501.13350*, 2025.

Yaniv Romano, Evan Patterson, and Emmanuel Candes. Conformalized quantile regression. *Advances in neural information processing systems*, 32, 2019.

Anna Sokol, Nuno Moniz, and Nitesh V Chawla. Conformalized selective regression. *Discover Data*, 4(1):14, 2026.

David Stutz, Krishnamurthy Dj Dvijotham, Ali Taylan Cemgil, and Arnaud Doucet. Learning optimal conformal classifiers. In *International Conference on Learning Representations*, 2022.

Xiaoyi Su, Zhixin Zhou, and Rui Luo. Co-optimization for adaptive conformal prediction. *arXiv preprint arXiv:2603.01719*, 2026.

Makoto Takamoto, Timothy Praditia, Raphael Leiteritz, Daniel MacKinlay, Francesco Alesiani, Dirk Pflüger, and Mathias Niepert. Pdebench: An extensive benchmark for scientific machine learning. *Advances in neural information processing systems*, 35:1596–1611, 2022.

Iqtedar Uddin, Mazin Khider, and André Bauer. Proactive routing to interpretable surrogates with distribution-free safety guarantees. *arXiv preprint arXiv:2603.14623*, 2026.

Nobuyuki Umetani and Bernd Bickel. Learning three-dimensional flow for interactive aerodynamic design. *ACM Transactions on Graphics (TOG)*, 37(4):1–10, 2018.

Vladimir Vovk, Alexander Gammerman, and Glenn Shafer. *Algorithmic learning in a random world*. Springer, 2005.

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

Ran Xie, Rina F Barber, and Emmanuel J Candès. Boosted conformal prediction intervals. *Advances in Neural Information Processing Systems*, 37:71868–71899, 2024.

Christopher Yeh, Nicolas Christianson, Alan Wu, Adam Wierman, and Yisong Yue. End-to-end conformal calibration for optimization under uncertainty. *Transactions on Machine Learning Research*, 2025. ISSN 2835-8856.

A Extended Related Work

This section expands the comparison in Section 2 and distinguishes the statistical targets and modeling objectives of the most closely related approaches.

Selective prediction and model routing. Selective prediction allows a model to withhold its prediction when it is unreliable, optionally deferring to a fallback model or expert. Learning-to-defer methods jointly learn when to predict and when to delegate to a downstream expert, accounting for both model and expert performance in the system objective (Madras et al., 2018; Mozannar & Sontag, 2020). Conformalized selective regression instead uses conformalized interval width as an input-dependent rejection score (Sokol et al., 2026). More generally, deep ensembles (Lakshminarayanan et al., 2017), MC-Dropout (Gal & Ghahramani, 2016), and evidential regression (Amini et al., 2020) provide uncertainty estimates that can serve as deferral scores when incorporated into a routing system. Learning-to-defer methods optimize the system-level handoff but do not construct conformal intervals, whereas conformalized selective regression and the uncertainty-quantification methods above use generic uncertainty measures for rejection. None of these approaches trains conformal interval allocation specifically around a downstream decision threshold.

Most closely related to our setting, Uddin et al. (2026) train an input-based gate that chooses between an interpretable surrogate and a reference model before either model is evaluated. They train the gate to distinguish safe from unsafe inputs and calibrate its routing threshold on held-out data to ensure, with probability at least $1 - \delta$, that the violation rate among inputs routed to the surrogate is at most α. This guarantee concerns performance degradation conditional on routing, whereas our conformal interval provides marginal coverage for the simulator-derived decision quantity. Their method also does not learn how the interval width should be allocated around a downstream specification threshold.

Conformal efficiency optimization. Split conformal prediction (CP) provides finite-sample marginal coverage under exchangeability (Vovk et al., 2005; Papadopoulos et al., 2002); Angelopoulos & Bates (2023) give a modern introduction. Conformalized Quantile Regression (CQR) adapts interval width to heteroscedasticity by conformalizing learned conditional quantiles (Romano et al., 2019). Other methods optimize conformal efficiency more directly. ConfTr differentiates through a mini-batch approximation to conformal prediction to optimize the size and composition of classification sets (Stutz et al., 2022), while VR-ConfTr reduces the variance of its gradient estimates (Noorani et al., 2026). For regression, DOICR directly minimizes average interval width (Lei & Bellotti, 2023); Boosted Conformal Intervals uses gradient boosting to refine a predefined conformity score for shorter intervals or improved conditional coverage (Xie et al., 2024); and CoCP jointly learns interval centers and radii (Su et al., 2026). These methods optimize conformal efficiency or other interval-quality objectives, but their objectives are not localized by distance to an externally specified downstream decision threshold.

Decision-aware conformal prediction. A growing line of work connects conformal methods with downstream decisions. Conformal Decision Theory calibrates decisions directly to control their realized average loss (decision risk), without first constructing prediction sets (Lekeufack et al., 2024). Utility-Directed CP incorporates a downstream cost function into the construction of classification prediction sets while retaining standard coverage guarantees (Cortes-Gomez et al., 2025). End-to-end conformal calibration for optimization learns uncertainty sets for conditional robust optimization using the downstream decision loss (Yeh et al., 2025). These methods show how conformal procedures can account for decision risk or utility, but they do not study when to invoke a high-fidelity simulator according to whether a scalar conformal interval crosses a fixed specification threshold.

Scientific surrogates and calibrated uncertainty. Scientific surrogate modeling spans neural operators such as FNO (Li et al., 2021) and GINO (Li et al., 2023), graph-based simulators such as MeshGraphNet (Pfaff et al., 2021) and X-MeshGraphNet (Nabian et al., 2024), and transformer-based models such as Transolver (Wu et al., 2024), Transolver++ (Luo et al., 2025), GeoTransolver (Adams et al., 2025), and AB-UPT (Alkin et al., 2025). Complementary work studies uncertainty quantification and calibration for these surrogates. Magnani et al. (2025) develop an approximate Bayesian framework for uncertainty quantification in neural operators. Mouli et al. (2024) evaluate several uncertainty-quantification methods for neural operators under out-of-domain inputs and introduce DiverseNO, a multi-head model designed to approximate the uncertainty benefits of an ensemble at lower computational cost. Ma et al. (2024) provide distribution-free, finite-sample cal-

ibration for function-valued operator outputs, targeting the expected fraction of domain points covered by the uncertainty band, and Qin et al. (2024) use surrogate models and conformal inference to provide probabilistic guarantees for the satisfaction of formal system specifications. These methods quantify or calibrate surrogate outputs and formal properties, but they do not learn threshold-local conformal interval allocation for selective simulator invocation.

Table 1 condenses these distinctions. Because the cited methods establish different statistical statements, a checkmark in the guarantee column indicates the presence of a distribution-free coverage or risk guarantee, not that the guaranteed quantities are equivalent.

B Additional decision-level supervision diagnostic

The comparison with Uniform in Section 4.2 is our primary evidence for separating the benefit of threshold-local weighting from changes in predictor quality. As an additional check, Table 7 augments each baseline with an extra unweighted g-level MSE loss while leaving TACR without this auxiliary term. On both CircuitNet and AirfRANS, TACR routes fewer samples than every baseline in this comparison. Thus, adding this direct supervision of the decision quantity does not by itself reproduce the routing efficiency of TACR on these two datasets.

Table 7: Explicit decision-level supervision does not reproduce TACR's routing gains: without auxiliary g-MSE, TACR attains the lowest mean SIM% on both datasets. Values are percentages reported as mean±std over five seeds; bold marks the lowest mean.

Method	CircuitNet	AirfRANS
Standard CP	18.5±0.6	11.9±0.8
CQR	37.5±2.1	9.8±1.0
Uniform	12.2±0.8	8.5±0.8
TW-UQ	21.7±3.1	9.7±1.0
TACR (no aux)	**11.6±0.5**	**7.4±0.9**

C Full per-dataset metrics

Table 8 summarizes the core notation and gives plain-language interpretations of the reported metrics. SIM% and MISS% form the primary cost–risk pair, whereas g-MAE and NEEDS% diagnose predictor accuracy and the surrogate-specific zero-miss routing reference, respectively.

Table 9 reports the complete metric set for every method on every dataset (mean over five seeds for non-ensemble methods; ensemble single run). Precision, recall, and F1 score (Prec, Rec, and F1) evaluate routing as a detector of the NEEDS% event, and ρ is the Spearman correlation between σ and absolute prediction error. The training objective does not directly optimize this correlation.

D Conditional behavior near τ

Although split conformal calibration controls marginal noncoverage, routing depends on where that noncoverage occurs. Near τ, tighter intervals are less likely to cross the threshold and can reduce SIM%, particularly when the predicted decision quantity is accurate; noncoverage there, however, is also the most likely to produce a wrong unrouted decision. Far from τ, noncoverage is often decision-benign because $g(y)$ can fall outside the interval while remaining on the same side of the threshold as $\hat{g}$. Thus, allocating noncoverage to near-threshold or far-threshold inputs can respectively improve routing efficiency or use the error budget where it is less likely to increase MISS%. Figure 5 examines this behavior by stratifying test samples according to either the predicted margin $|\hat{g} - \tau|$ or the ground-truth distance $|g(y) - \tau|$.

The bars show that noncoverage is not distributed uniformly across distance bands: it is often elevated near the threshold or in the farthest bands relative to intermediate distances. Misses, by contrast, are concentrated near τ and vanish in most farther bands. Darcy provides the clearest example of the second effect: far-band noncoverage reaches approximately 65–80%, yet no misses

Table 8: Core notation and evaluation metrics. Each quantity is paired with its operational role in conformal routing; percentages are computed over test inputs.

Notation	Definition	Description
$\hat{g}(x)$	$g(f_\theta(x))$	Surrogate estimate of the scalar decision quantity used for routing.
τ	Fixed decision threshold	Application-specified limit against which the decision quantity is compared, such as a maximum allowable stress or a minimum required lift coefficient.
$\sigma(x)$	Positive input-dependent scale	Learned relative-width factor that allows interval width to vary across inputs.
$\hat{q}$	Calibration multiplier in Eq. 12	Value computed from held-out calibration scores to obtain the conformal coverage guarantee.
$C(x)$	$[\hat{g}(x) - \hat{q}\sigma(x), \hat{g}(x) + \hat{q}\sigma(x)]$	Calibrated prediction interval for the scalar decision quantity.
SIM%	$100\Pr(r(X) = 1)$	Inputs that still require a high-fidelity simulator call; the primary cost metric.
MISS%	$100\Pr(\hat{d}(X) \neq d(Y), r(X) = 0)$	Wrong surrogate threshold decisions that escape simulation; the accompanying risk metric.
g-MAE	$\mathbb{E}[\lvert\hat{g}(X) - g(Y)\rvert]$	Average error in the scalar decision quantity; a diagnostic of predictor accuracy.
NEEDS%	$100\Pr(\hat{d}(X) \neq d(Y))$	Inputs the surrogate alone would classify incorrectly; for a fixed surrogate, the zero-miss oracle routing rate.

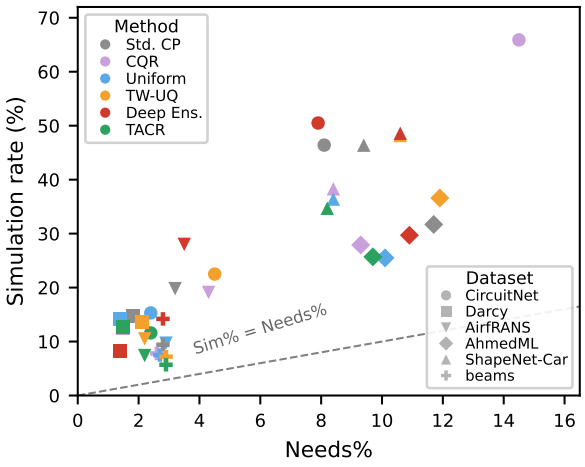

Figure 4: **Routing relative to the zero-miss reference across all methods.** The dashed line, SIM% = NEEDS%, marks the minimum routing rate under zero-miss oracle routing for each fixed surrogate. Color denotes method, and marker shape denotes dataset; "Std. CP" and "Deep Ens." denote Standard CP and Deep Ensemble.

are observed there because the uncovered targets remain on the correct side of the threshold. This separation between noncoverage and decision error is consistent with the intended allocation of interval width: the resulting allocation can trade local coverage behavior for reduced routing where the decision is difficult, while noncoverage far from τ is comparatively inexpensive in decision risk. These stratified results are diagnostic rather than a conditional guarantee; applications requiring coverage within prespecified regions would need group-conditional or another locally valid calibration procedure.

Table 9: Uniform and TACR have closely matched g-MAE and Needs% across datasets, making their routing differences informative about threshold-local interval allocation rather than predictor quality. Prec, Rec, and F1 denote routing precision, recall, and F1 score. Needs, SIM%, and MISS% are percentages, whereas Prec, Rec, F1, and ρ are fractions. Bold marks the best value within each dataset and metric: lower for g-MAE, Needs, SIM%, and MISS%, and higher for Prec, Rec, F1, and ρ. Displayed ties are all bolded.

Dataset	Method	g-MAE	Needs%	SIM%	Prec	Rec	F1	MISS%	ρ
CircuitNet	Standard CP	0.0082	8.1	46.4	0.17	0.95	0.28	0.37	0.00
	CQR	0.0127	14.5	65.9	**0.22**	**0.99**	**0.36**	**0.09**	-0.06
	Uniform	**0.0023**	**2.4**	15.3	0.14	0.90	0.24	0.23	0.18
	TW-UQ	0.0053	4.5	22.5	0.19	0.97	0.32	0.12	**0.66**
	Deep Ensemble	0.0083	7.9	50.5	0.15	**0.99**	0.27	0.12	0.29
	TACR	0.0024	**2.4**	**11.6**	0.17	0.82	0.28	0.44	0.22
Darcy	Standard CP	0.0135	1.8	14.8	0.12	0.98	0.21	0.03	0.00
	CQR	0.0131	1.5	12.5	0.14	0.93	0.23	0.11	0.60
	Uniform	0.0129	**1.4**	14.1	0.09	0.99	0.17	0.02	-0.06
	TW-UQ	0.0152	2.1	13.6	0.14	0.94	0.25	0.13	0.43
	Deep Ensemble	**0.0127**	**1.4**	**8.3**	**0.17**	**1.00**	**0.29**	**0.00**	**0.70**
	TACR	0.0129	1.5	12.7	0.11	0.96	0.20	0.06	-0.31
AirfRANS	Standard CP	0.0777	3.2	19.8	0.15	0.95	0.26	0.20	0.00
	CQR	0.0678	4.3	19.1	0.21	0.93	0.34	0.30	**0.52**
	Uniform	**0.0368**	2.9	9.7	**0.29**	0.95	**0.44**	0.10	0.25
	TW-UQ	0.0485	**2.2**	10.5	0.17	0.68	0.27	0.40	0.25
	Deep Ensemble	0.0694	3.5	28.0	0.12	**1.00**	0.22	**0.00**	0.28
	TACR	0.0369	**2.2**	**7.4**	0.25	0.81	0.38	0.40	0.14
AhmedML	Standard CP	0.0172	11.7	31.7	**0.31**	0.83	**0.45**	1.98	0.00
	CQR	0.0153	**9.3**	27.9	0.27	0.83	0.40	**1.78**	0.06
	Uniform	0.0149	10.1	**25.5**	**0.31**	0.74	0.43	2.57	0.00
	TW-UQ	0.0208	11.9	36.6	0.28	**0.85**	0.42	**1.78**	**0.33**
	Deep Ensemble	0.0190	10.9	29.7	0.27	0.73	0.39	2.97	0.18
	TACR	**0.0148**	9.7	25.7	**0.31**	0.75	0.42	2.38	0.00
ShapeNet-Car	Standard CP	0.0132	9.4	46.4	0.19	0.92	0.31	0.78	0.00
	CQR	0.0110	8.4	38.3	0.20	0.92	0.33	0.67	0.00
	Uniform	0.0101	8.4	36.4	**0.22**	0.93	**0.35**	0.56	0.00
	TW-UQ	0.0140	10.6	48.2	0.21	**0.96**	**0.35**	**0.45**	**0.21**
	Deep Ensemble	0.0129	10.6	48.6	0.21	0.95	0.34	0.56	0.20
	TACR	**0.0095**	**8.2**	**34.7**	**0.22**	0.92	**0.35**	0.56	0.00
Beams	Standard CP	0.1314	2.8	9.4	0.28	**0.93**	0.43	**0.20**	0.00
	CQR	0.1302	**2.6**	7.8	0.31	0.92	0.46	**0.20**	**0.55**
	Uniform	0.1320	2.7	7.5	0.32	0.87	0.46	0.35	**0.55**
	TW-UQ	0.1329	2.9	7.2	0.35	0.86	0.49	0.40	**0.55**
	Deep Ensemble	**0.1286**	2.8	14.2	0.18	0.91	0.29	0.25	0.37
	TACR	0.1336	2.9	**5.7**	**0.40**	0.79	**0.53**	0.60	0.44

E IMPLEMENTATION DETAILS

Data splits. Table 10 reports the exact splits used in all finalized experiments. Each dataset is partitioned into training, calibration, and test sets, with no separate validation set. The split is generated once with seed 42 and held fixed across run seeds, so repeated runs differ only in model initialization and training stochasticity rather than data membership. Models are trained for fixed epoch counts and evaluated at the final epoch, without early stopping or checkpoint selection. The calibration partition is reserved exclusively for conformal calibration.

Table 10: Exact data splits used in the experiments. A dash indicates that no separate validation partition is used.

Dataset	Total	Train	Validation	Calibration	Test
CircuitNet	3,456	2,073	–	519	864
Darcy	10,000	6,000	–	2,000	2,000
AirfRANS	1,000	600	–	200	200
AhmedML	498	298	–	99	101
ShapeNet-Car	888	532	–	177	179
Beams	2,000	1,200	–	400	400

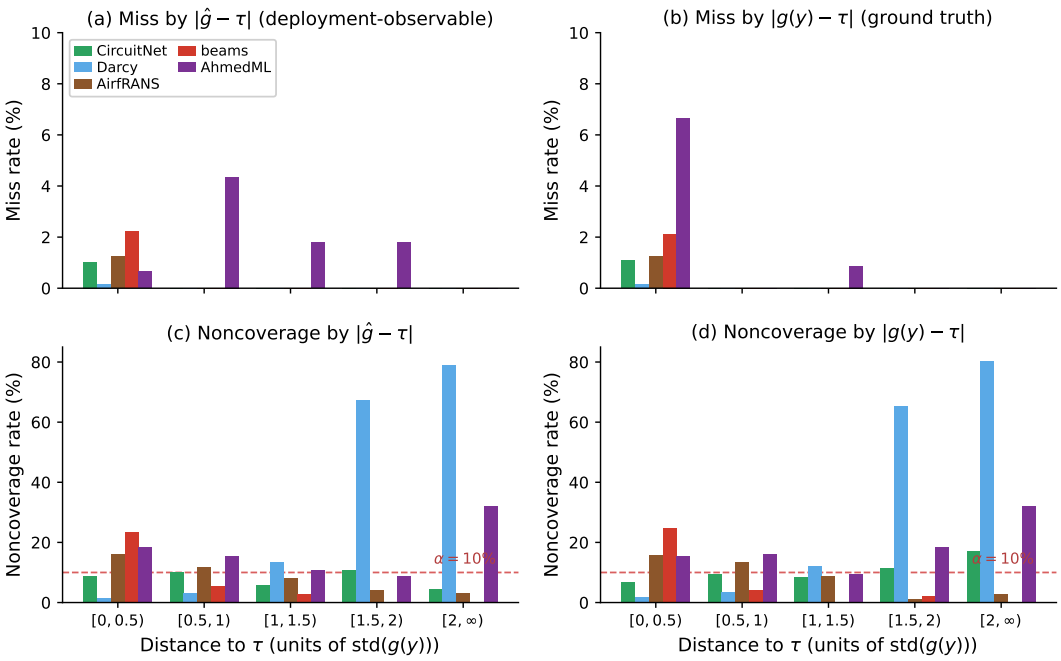

Figure 5: **Distance-stratified misses and noncoverage for TACR.** Bars pool five seeds for each displayed dataset. The top row reports miss and the bottom row noncoverage ($100\% - \text{coverage}$); columns bin by the predicted margin (left) or ground-truth distance (right), normalized by the standard deviation of $g(y)$. The dashed line marks the marginal $\alpha = 0.1$ noncoverage target, not a bandwise constraint.

Dataset preprocessing. No cases are filtered from CircuitNet or Darcy. Darcy's inputs and targets are standardized. AirfRANS subsamples 5,000 surface and volume points per geometry and constructs radius graphs with radius 0.05 and at most 64 neighbors. AhmedML uses 48,000 points per geometry during training and 64,000 during calibration and testing. ShapeNet-Car uses point coordinates and surface normals, standardized across the dataset. The beams dataset is generated with seed 42 and contains five input features.

Backbones and scale heads. Table 11 summarizes the dataset-specific architectures and optimization settings. CircuitNet uses ResNet18-UNet++ with its encoder initialized from ImageNet weights; all other backbones are initialized from scratch. AirfRANS uses three GraphSAGE layers with input dimension 7 and hidden, latent, and scale-projection dimensions of 64. Beams uses a three-layer MLP with width 64. AhmedML and ShapeNet-Car use GeoTransolver with hidden dimension 160, 20 layers, 128 slices, and eight heads.

For Uniform, TW-UQ, and TACR, we parameterize $\sigma(x) = \sigma_\phi(h_\theta(x))$, where $h_\theta(x)$ is the predictor's penultimate backbone representation. The scale head projects the spatial, node, or latent features to a lower-dimensional representation, pools them into one vector per sample, and applies a small MLP with a Softplus output to ensure positivity. It attaches to the encoder bottleneck for convolutional models, pools node or point features for graph- and transformer-based models, and uses the shared MLP trunk for beams. Thus, the scale head reuses features already computed for prediction and introduces no separate feature extractor.

Prediction objectives. For Standard CP, Uniform, and TACR, the prediction objective is mean-squared error on the predicted field for CircuitNet, Darcy, AirfRANS, AhmedML, and ShapeNet-Car, and mean-squared error on the scalar $\log_{10}$ peak-stress target for beams. CQR replaces the standard prediction head with two conditional-quantile outputs and optimizes the pinball loss. Each Deep Ensemble member uses the corresponding dataset-level prediction objective.

TW-UQ objective. This baseline augments the dataset-level prediction loss with a threshold-weighted error on the scalar decision quantity and a Gaussian negative-log-likelihood objective for

Table 11: Dataset-specific configurations. Batch sizes are shown as training/calibration/test. E_1 and E_2 denote the two stages used by Uniform, TW-UQ, and TACR; single-stage methods train for E_1 epochs. All optimizers use weight decay 10^{-4} and cosine learning-rate decay within each stage. For AhmedML, the two learning rates denote TACR and the baselines, respectively.

Dataset	Backbone	Batch	Optimizer	Base LR	E_1/E_2
CircuitNet	ResNet18-UNet++	16/32/32	AdamW	10^{-4}	200/100
Darcy	5-level U-Net	32/64/64	AdamW	10^{-3}	500/250
AirfRANS	3-layer GraphSAGE	16/16/16	AdamW	10^{-3}	200/100
AhmedML	GeoTransolver	4/2/2	Adam	$10^{-3}/5\times10^{-4}$	300/150
ShapeNet-Car	GeoTransolver	4/2/2	Adam	5×10^{-4}	300/150
Beams	3-layer MLP	64/256/256	AdamW	10^{-3}	200/100

a positive scale σ_i produced by a Softplus head. Let $g_i = g(y_i)$ and $\epsilon = 10^{-8}$, and define

$$w_i^{\mathrm{TW}} = 1 + 2\exp\left(-\frac{(\mathrm{sg}[\hat{g}_i] - \tau)^2}{2\,\mathrm{bw}^2}\right), \tag{15}$$

where sg denotes stop-gradient. During joint training, the minibatch objective is

$$\mathcal{L}_{\text{TW-UQ}} = \mathcal{L}_{\mathrm{pred}} + \tfrac{1}{2}\mathcal{L}_{g,\mathrm{IW}} + \tfrac{1}{2}\mathcal{L}_{\mathrm{NLL}}, \tag{16}$$

$$\mathcal{L}_{g,\mathrm{IW}} = \frac{1}{|\mathcal{B}|}\sum_{i\in\mathcal{B}} w_i^{\mathrm{TW}}(\hat{g}_i - g_i)^2, \tag{17}$$

$$\mathcal{L}_{\mathrm{NLL}} = \frac{1}{|\mathcal{B}|}\sum_{i\in\mathcal{B}}\left[\log(\sigma_i+\epsilon) + \frac{\mathrm{sg}[g_i - \hat{g}_i]^2}{2\sigma_i^2 + \epsilon}\right]. \tag{18}$$

The stopped residual prevents the likelihood term from directly updating the mean prediction, while the weighted scalar error emphasizes prediction accuracy near τ. The predictor-only warm-up uses $\mathcal{L}_{\mathrm{pred}}$ alone. At calibration and deployment, TW-UQ uses $s_i = |g_i - \hat{g}_i|/(\sigma_i + \epsilon)$ and the same split-conformal interval $[\hat{g}_i - \hat{q}\sigma_i, \hat{g}_i + \hat{q}\sigma_i]$ as the other learned-scale methods. The objective and its coefficients are shared across all six datasets.

Decision quantities on the CFD datasets. For AirfRANS, AhmedML, and ShapeNet-Car, the routing quantity is computed by integrating the predicted surface pressure. AirfRANS uses the lift coefficient C_L, whereas AhmedML and ShapeNet-Car use the drag coefficient C_D. ShapeNet-Car additionally provides surface normals, which are required for the drag calculation.

Training-time soft quantile. Within each training mini-batch, $\mathrm{SoftQuantile}$ is implemented as a linearly interpolated empirical quantile of the normalized scores. After sorting the $m = |\mathcal{B}|$ scores, we evaluate the quantile at the continuous zero-based rank $(1-\alpha)(m-1)$ by interpolating between the adjacent order statistics. This approximation is used only during optimization and is distinct from the exact rank-based calibration rule in Eq. 12.

Conformal calibration. We use $\alpha = 0.1$, corresponding to 90% target marginal coverage. After training, the conformal multiplier is computed on the held-out calibration set using the exact order statistic in Eq. 12. The calibration set is disjoint from both training and testing and is not used for model fitting or for the training-time soft quantile.

Hyperparameter selection. All routing-objective hyperparameters are fixed globally rather than tuned separately for individual datasets. We use $\lambda = 0.5$, $\beta = 16$, and $\mu = 10^{-2}$ for every dataset and seed, and set $\mathrm{bw} = 0.3 \cdot \mathrm{std}_{\mathrm{train}}(g(y))$, where the standard deviation is computed over the training labels. These values are heuristic defaults and are not selected through dataset-specific search or by calibration- or test-set performance. The β sweep in Table 5 is reported only as a sensitivity analysis and is not used to select the default setting. Uniform follows the same architecture, scale-head parameterization, routing objective, and training schedule as TACR, with the sole change $\beta = 0$ to remove threshold-local weighting.

Table 12 summarizes the settings shared across the main experiments.

Table 12: Shared settings for the main experiments. The loss coefficients describe TACR, and Uniform uses the same routing objective with $\beta = 0$. The two-stage schedule applies to Uniform, TW-UQ, and TACR. Let E and η denote the dataset-specific base epoch count and learning rate.

Setting	**Value**
Conformal miscoverage level α	0.1
Decision threshold τ	Training-set 70th percentile of $g(y)$
Routing-loss weight λ	0.5
Threshold-local weighting strength β	16
Scale-regularization weight μ	0.01
Bandwidth bw	$0.3 \cdot \mathrm{std}_{\mathrm{train}}(g(y))$ (population std)
Stage 1 duration / predictor learning rate	E / η
Stage 2 duration	$E/2$
Stage 2 predictor / scale-head learning rates	0.1η / η
Evaluation epoch	Final epoch
Non-ensemble random seeds	$42, 123, 456, 789, 1011$
Deep Ensemble members	5 (one ensemble run)

Training schedules. Uniform, TW-UQ, and TACR use the same two-stage schedule. Stage 1 optimizes the predictor alone for the dataset-specific base duration E. Stage 2 jointly fine-tunes the predictor and the method-specific scale head for $E/2$ epochs, using one tenth of the base learning rate for predictor parameters and the base learning rate for scale-head parameters.

Standard CP and CQR use a single stage of E epochs. CQR replaces the prediction head with two conditional-quantile outputs and optimizes the pinball loss. Each Deep Ensemble member is trained independently for E epochs, using member seeds 100–104. Standard CP uses absolute residuals as nonconformity scores, CQR conformalizes its learned quantile interval, and Uniform, TW-UQ, and TACR use the positive Softplus scale head described above.