Neural Statistical Functions

Daniel Xu[1], **Yuxin Xie**[2], **Minghao Guo**[2✉], **Haixu Wu**[2✉], **Wojciech Matusik**[2]
[1]Columbia University, [2]MIT CSAIL
xu.daniel@columbia.edu, yxie25@mit.edu
{guomh2014,wuhaixu98}@gmail.com, wojciech@csail.mit.edu

Abstract

Neural networks typically predict responses at individual operating conditions, whereas many real-world tasks involving uncertainty or extreme-value analysis require statistics over a range of possible conditions. Estimating these quantities through repeated inference can incur substantial latency. We introduce *neural statistical functions*, a new family of models that directly infer statistics over continuous operating condition ranges without sampling at inference. By introducing the notion of *prefix statistics*, we transform and unify diverse statistical functions (*e.g.*, integrals, quantiles, and maxima) into an interval-conditional framework. An identity connecting prefix statistics to individual-case responses yields a learning objective that enables training from pretrained predictors and scattered observations without requiring statistical labels for entire intervals. Neural statistical functions achieve strong performance in estimating essential statistics of complex physical processes, including mean energy in dynamical systems, quantiles of aerodynamic responses, and maximum stress in crash processes, while achieving up to a 200× reduction in inference latency.

1 Introduction

Deep learning has become widely used to predict the behavior of complex systems [1, 8, 35, 15, 33]. In physics simulation, for example, neural surrogates predict physical responses—such as pressure, stress, or energy—from a geometry design and an operating condition [23, 34, 35]. While neural surrogates have substantially accelerated scientific discovery and engineering [34, 22, 2], the standard interface remains single-condition prediction: the model is queried at one specified condition and returns the corresponding response. However, many deployment-time queries are not single-condition predictions. Scientific discovery and engineering design often require understanding how a system behaves across a range of conditions, including its typical responses, variability, and extremes. These questions call for statistics such as mean energy over a time window, failure probabilities under uncertain loads, or maximum stress over impact angles. A standard approach is Monte Carlo sampling or dense sweeping: repeatedly evaluate the simulator or its neural surrogate at many conditions, then compute the desired statistic, which can become prohibitively expensive [3, 5] for numerical solvers and remain non-negligible for neural surrogates that output large spatial fields [36].

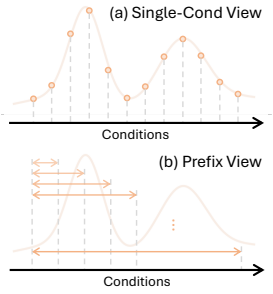

Figure 1: This paper adopts a new *prefix view* for statistics.

Consider industrial crash-test design as an example. An engineer may want to know the maximum stress experienced by a vehicle across a range of possible impact angles. A conventional neural surrogate predicts stress at each selected angle, so estimating this maximum requires repeated

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

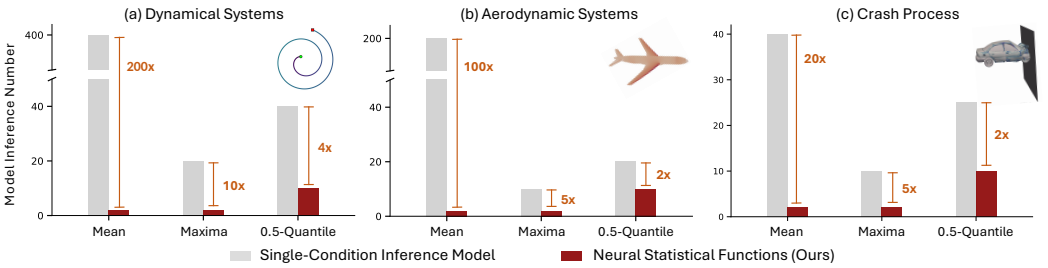

Figure 2: Number of model inferences to achieve comparable accuracy under three experimental scenarios. Compared to the single-condition inference model, neural statistical functions do not rely on unwieldy sampling operations, thereby significantly reducing the number of model inferences.

model evaluations and aggregation of the resulting predictions. More generally, let $\mathbf{x} \in \mathbb{R}^{N\times d}$ denote the spatial, geometric, or state representation at which a quantity of interest is evaluated (*e.g.*, vehicle geometry), c the operating condition (*e.g.*, crash impact angle), and $\boldsymbol{h}(\mathbf{x}, c)$ the response of interest (*e.g.*, inner stress). Our goal is to infer a statistic of the response over any interval $[c_0, c_1] \subseteq [c_{\min}, c_{\max}]$:

$$\boldsymbol{h}_{\text{Stat}}(\mathbf{x}, [c_0, c_1]) = \text{Stat}\left(\{\boldsymbol{h}(\mathbf{x}, c); c \sim [c_0, c_1] \subseteq [c_{\min}, c_{\max}]\}\right), \tag{1}$$

Here, Stat may represent an integral, quantile, maximum, or another statistic. Under the single-condition approach in Figure 1(a), a neural network $\boldsymbol{h}_\theta \approx \boldsymbol{h}$ predicts individual responses [19, 34], which are then aggregated to estimate the requested statistic. However, this single-condition approach incurs high computational costs through repeated model evaluations, and cannot easily adapt to changes to the condition interval which require a new round of sampling.

We introduce *neural statistical functions*, a new family of deep learning models that directly infer statistics over continuous operating condition ranges. We depart from this convention by adopting a new perspective on the varying-condition regression task, which we call the *prefix* view. As demonstrated in Figure 1(b), rather than learning the specific quantity under a single operating condition c, we propose to learn the neural network to approximate the *prefix statistics* $\tilde{\boldsymbol{h}}(\mathbf{x}, \cdot)$. Taking integral estimation as an example, let $\tilde{\boldsymbol{h}}(\mathbf{x}, c)$ denote the accumulated response from $c_{\min}$ to c. Then the integral over any interval $[c_0, c_1]$ can be recovered by simple subtraction, which can be formally described as:

$$\boldsymbol{h}_{\text{integral}}(\mathbf{x}, [c_0, c_1]) = \tilde{\boldsymbol{h}}_{\text{integral}}(\mathbf{x}, c_1) - \tilde{\boldsymbol{h}}_{\text{integral}}(\mathbf{x}, c_0), \text{ where } \tilde{\boldsymbol{h}}_{\text{integral}}(\mathbf{x}, c) = \int_{c_{\min}}^{c} \boldsymbol{h}(\mathbf{x}, u)\mathrm{d}u. \tag{2}$$

This parameterization uses the nonlinear approximation capacity of neural networks to learn the cumulative response $\tilde{\boldsymbol{h}}(\cdot, \cdot)$ directly, enabling statistical inference over varying condition intervals without explicit sampling at inference time.

Introducing the prefix view does not, however, by itself, ensure successful training and adoption of neural statistical functions for general statistical quantities. In addition to introducing this new type of neural network, we also construct the whole training and inference pipeline. **1)** Regarding inference generalizability, although the prefix view is natural for integrals, it is non-trivial to extend this idea to more complex statistics, *e.g.*, quantiles and maxima, whose underlying statistical functions are not additive over intervals. Through a set of principled mathematical transformations, we demonstrate that these widely used statistics can be accurately calculated or approximated based on the prefix integral for a transformed function, which enables the general use of neural statistical functions. **2)** Regarding training efficiency, rather than directly using the prefix integral computed from ground-truth data as supervision, which may also be extremely time-consuming and biased due to data sparsity, we find a principled identity that connects prefix statistics and the individual-case regression model $\boldsymbol{h}(\cdot, \cdot)$, and further derive an efficient and stable learning objective.

Our theoretical analysis connects local supervision to statistical prediction error and characterizes how query resolution governs the representation requirements of statistical inference (Section 3). We evaluate neural statistical functions on three physical applications: mean energy in dynamical systems, pressure quantiles in aerodynamics, and maximum stress in crash processes. Across these

tasks, neural statistical functions accurately estimate interval statistics with up to $200\times$ lower inference latency. Additional comparisons with functional-output predictors, joint angle-of-attack–Mach queries, and hybrid-supervision ablations further support the proposed inference and training mechanisms (Appendices B.1–B.2 and Section 4.3). Together, these results demonstrate the potential of neural statistical functions to accelerate scientific discovery and engineering design through efficient statistical analysis under varying operating conditions.

2 Related work

As a new type of neural network, neural statistical functions define a complete training–inference paradigm. While technically distinct, they are conceptually related to several lines of prior work.

Integrals in neural networks Integrals appear widely in neural networks as a basic operation in model architectures [10, 11] and generative models [14, 21, 31, 13]. Neural ODE [10] reformulates the representation learning in multilayer neural networks as a continuous-time integral, where the neural network learns the derivative of hidden states. Extending the numerical integral from model depth to real-world time, Neural ODE can be applied to simulate dynamic processes [27]. Recently, diffusion models have been developed based on multi-step denoising, where the numerical integral is adopted for inference [14, 21]. Specifically, Mean Flow [13] proposes to directly learn the average velocity with one neural network, which does not rely on numerical integration along the denoising steps. Unlike prior work, our models focus on the statistical estimation over varying conditions and present a general framework for statistics beyond integral.

Stochastic analysis Mechanical design often relies on stochastic analysis to evaluate system behavior under uncertain operating conditions. Classical approaches such as stochastic finite element method (FEM) [32, 18] provide principled tools for such analysis, but they can be computationally expensive despite developments in acceleration techniques. Although neural networks have been developed as efficient surrogates for numerical methods [17, 6], prior work focuses on deterministic methods, where neural networks for stochastic analysis, *e.g.*, stochastic FEM, are still underexplored. Uncertainty quantification for neural networks is also related, as it typically estimates the uncertainty or variance of model predictions for a fixed input [12]. In contrast, this paper focuses on statistical properties under varying conditions, rather than uncertainty at a single condition.

3 Neural Statistical Functions

As formalized in Eq. (1), given that $\mathrm{Stat}(\cdot)$ specifies the type of statistic and $\boldsymbol{h}(\mathbf{x}, c)$ denotes the target quantity at position $\mathbf{x}$ under condition c, our goal is to predict the statistics $\boldsymbol{h}_{\text{Stat}}(\mathbf{x}, [c_0, c_1])$ corresponding to the condition interval $[c_0, c_1] \subseteq [c_{\min}, c_{\max}] \subset \mathbb{R}$. We first develop the framework for a scalar condition, for which users may query arbitrary intervals, and subsequently evaluate its extension to two-dimensional box queries in Appendix B.2.

Next, we will first extend the prefix view in Eq. (2) to general statistics inference, such as quantile and maxima, where a *prefix statistics framework* is presented. Then, we will derive the training objective of neural statistical functions, which is based on an identity principle of prefix statistics. We also connect local training residuals to interval prediction error and discuss the implications for statistical representation, with the formal compression analysis provided in Appendix A.4. For main text clarity, we defer all the proofs of propositions and theorems to the Appendix A.

Notation simplification For clarity, we reparameterize the scalar uncertainty in quantile space. We assume μ is atomless and consider only intervals of positive probability mass. Let

$$s = Q_\mu(c) \in [0, 1], \qquad c = Q_\mu^{-1}(s), \tag{3}$$

where Q_μ is the cumulative distribution function of the condition c's distribution μ and Q_μ^{-1} is its generalized inverse quantile map. Under this change of variables, any interval $[c_0, c_1]$ induces a quantile interval

$$s_0 := Q_\mu(c_0), \qquad s_1 := Q_\mu(c_1), \qquad m := s_1 - s_0. \tag{4}$$

The quantity m is the probability mass of the interval under μ. Quantile space is a convenient coordinate system because the condition distribution is mapped to the uniform distribution on $[0, 1]$,

and conditional integrals over $[c_0, c_1]$ become normalized integrals over $[s_0, s_1]$. This allows the learning problem to be expressed in a distribution-normalized way that is independent of the original scale of c.

3.1 Prefix statistics framework

To support general statistical functions, we built a framework based on the transformed local signal $\boldsymbol{\psi}(\mathbf{x}, s)$, whose choice depends on the target statistical function (Table 1).

For a given $\boldsymbol{\psi}$, define its prefix statistics by integration.

Table 1: Unified prefix-statistic framework. Each statistic is defined by a task-specific transformed local signal $\boldsymbol{\psi}$.

$\boldsymbol{h}_{\text{Stat}}(\mathbf{x}, [c_0, c_1])$	$\boldsymbol{\psi}(\mathbf{x}, s)$
Mean	$\boldsymbol{h}(\mathbf{x}, Q_\mu^{-1}(s))$
Max	$\exp\big(\beta\, \boldsymbol{h}(\mathbf{x}, Q_\mu^{-1}(s))\big)$
Quantile (CDF)	$\mathbb{1}\big\{\boldsymbol{h}(\mathbf{x}, Q_\mu^{-1}(s)) \le \mathbf{y}\big\}$

$$\tilde{\boldsymbol{h}}(\mathbf{x}, s) := \frac{1}{s}\int_0^s \boldsymbol{\psi}(\mathbf{x}, t)\,\mathrm{d}t, \qquad s \in (0, 1]. \tag{5}$$

This prefix-statistics construction defines a broader amortizable framework for interval-conditioned prediction. In particular, whenever a target statistic can be expressed as a function of a normalized interval integral,

$$\boldsymbol{h}_{\text{Stat}}(\mathbf{x}, [c_0, c_1]) = \boldsymbol{\Phi}\left(\frac{1}{m}\int_{s_0}^{s_1} \boldsymbol{\psi}(\mathbf{x}, s)\,\mathrm{d}s\right) = \boldsymbol{\Phi}\left(\frac{s_1\tilde{\boldsymbol{h}}(\mathbf{x}, s_1) - s_0\tilde{\boldsymbol{h}}(\mathbf{x}, s_0)}{m}\right),\ m := s_1 - s_0. \tag{6}$$

Thus, the proposed framework is a reusable primitive for interval-conditioned prediction. The interval mean and maximum are direct instances with different choices of the post-processing map $\boldsymbol{\Phi}$, while the quantile case fits this template at the level of the interval cumulative distribution, followed by inversion in the response variable. Next, we list the implementation for several typical statistics.

(i) Mean. Similar to the usage described in Eq. (2), the mean can be calculated by defining $\boldsymbol{\Phi}$ as the identity map.

(ii) Maxima. We adopt a smooth log-sum-exp surrogate to approximate the maxima function:

$$\boldsymbol{h}^{\beta}_{\text{Stat}}(\mathbf{x}, [c_0, c_1]) := \frac{1}{\beta}\log\left(\frac{1}{m}\int_{s_0}^{s_1} \exp\big(\beta\, h(\mathbf{x}, Q_\mu^{-1}(s))\big)\,\mathrm{d}s\right), \tag{7}$$

with temperature parameter $\beta > 0$, namely $\boldsymbol{\Phi}(\cdot) = \frac{1}{\beta}\log(\cdot)$. This is a continuous version of entropy-based smoothing for max-type objectives. As $\beta \to \infty$, the quantity converges to the interval supremum under mild regularity conditions [24].

(iii) α-Quantile. Here, we consider α-quantiles of different condition intervals, where $\alpha \in (0, 1)$ is a target probability level. Specifically, the interval-conditioned α-quantile of the response is defined by the generalized inverse of its conditional CDF:

$$\boldsymbol{h}^{\alpha}_{\text{Stat}}(\mathbf{x}, [c_0, c_1]) := \inf\left\{\mathbf{y} :\ \mathbb{P}(\boldsymbol{h}(\mathbf{x}, c) \le \mathbf{y} \,|\, c \in [c_0, c_1]) \ge \alpha\right\}. \tag{8}$$

Towards a unified framework, we work with the conditional cumulative distribution function (CDF) over response values $\boldsymbol{h}(\cdot, \cdot)$. Let $\mathbf{y}$ denote a candidate point-wise response level and define

$$\boldsymbol{\psi}(\mathbf{x}, s; \mathbf{y}) := \mathbb{1}\big\{\boldsymbol{h}(\mathbf{x}, Q_\mu^{-1}(s)) \le \mathbf{y}\big\},\ \tilde{\boldsymbol{h}}(\mathbf{x}, s; \mathbf{y}) := \frac{1}{s}\int_0^s \mathbb{1}\big\{\boldsymbol{h}(\mathbf{x}, Q_\mu^{-1}(t)) \le \mathbf{y}\big\}\,\mathrm{d}t. \tag{9}$$

Then, the interval-conditioned α-quantile can be obtained by inversion:

$$\boldsymbol{h}^{\alpha}_{\text{Stat}}(\mathbf{x}, [c_0, c_1]) := \inf\left\{\mathbf{y} :\ \frac{1}{m}\int_{s_0}^{s_1} \mathbb{1}\big\{\boldsymbol{h}(\mathbf{x}, Q_\mu^{-1}(t)) \le \mathbf{y}\big\}\,\mathrm{d}t \ge \alpha\right\}. \tag{10}$$

We recover quantiles by bisection over the response threshold. Reducing the search bracket to width ϵ requires $\mathcal{O}(\log((y_{\max} - y_{\min})/\epsilon))$ steps, each using two prefix evaluations.

Remark 1 (Generality of framework)**.** *Beyond α-quantile, Eq.* (9) *actually learns the interval-conditioned CDF of the response distribution, which can serve as a general primitive for a broad*

range of distributional statistics. For example, the probability that the response lies within a target range $[\mathbf{a}, \mathbf{b}]$ *can be directly computed as follows*

$$\begin{aligned}\mathbb{P}(\mathbf{a} < \boldsymbol{h}(\mathbf{x}, c) \leq \mathbf{b} \,|\, c \in [c_0, c_1]) = \frac{s_1 \tilde{\boldsymbol{h}}(\mathbf{x}, s_1; \mathbf{b}) - s_0 \tilde{\boldsymbol{h}}(\mathbf{x}, s_0; \mathbf{b})}{m} \\ - \frac{s_1 \tilde{\boldsymbol{h}}(\mathbf{x}, s_1; \mathbf{a}) - s_0 \tilde{\boldsymbol{h}}(\mathbf{x}, s_0; \mathbf{a})}{m}.\end{aligned} \tag{11}$$

Similarly, the probability of exceeding a critical threshold $\boldsymbol{\tau}$ *is given by*

$$\mathbb{P}(\boldsymbol{h}(\mathbf{x}, c) > \boldsymbol{\tau} \,|\, c \in [c_0, c_1]) = 1 - \frac{s_1 \tilde{\boldsymbol{h}}(\mathbf{x}, s_1; \boldsymbol{\tau}) - s_0 \tilde{\boldsymbol{h}}(\mathbf{x}, s_0; \boldsymbol{\tau})}{m}. \tag{12}$$

Therefore, although quantile inference requires an additional inversion step, the learned conditional CDF provides a reusable distributional representation for many downstream risk-aware statistics.

3.2 Model training

By introducing prefix statistics (Eq. (5)), we can set the statistics estimation free from computation-intensive sampling. Next, we will detail how to train a neural surrogate for $\tilde{\boldsymbol{h}}(\cdot, \cdot)$.

Prefix statistics identity Differentiating Eq. (5) with respect to s yields the first-order identity

$$s\, \partial_s \tilde{\boldsymbol{h}}(\mathbf{x}, s) + \tilde{\boldsymbol{h}}(\mathbf{x}, s) = \boldsymbol{\psi}(\mathbf{x}, s), \tag{13}$$

which directly relates the prefix statistic to the instantaneous solver response. It converts an interval-conditioned aggregation problem into a local differential constraint in quantiled condition space.

Proposition 2 (Equivalence of first order identity and prefix statistics)**.** *Fix* $\mathbf{x}$ *and suppose* $\boldsymbol{\psi}(\mathbf{x}, \cdot) \in L^1(0, 1]$. *Let* $\tilde{\boldsymbol{h}}(\mathbf{x}, \cdot)$ *be differentiable on* $(0, 1]$. *Then the following are equivalent:*

1. $\tilde{\boldsymbol{h}}$ *satisfies Eq.* (13) *a.e.* $s \in (0, 1]$ *together with the boundary condition* $\lim_{s \to 0} s\, \tilde{\boldsymbol{h}}(\mathbf{x}, s) = 0$*;*
2. $\tilde{\boldsymbol{h}}$ *admits the prefix-integral representation in Eq.* (5)*.*

Proposition 2 demonstrates that the first-order identity is not merely a property of the prefix statistic in Eq. (5); together with the natural boundary condition at $s \to 0$, it uniquely characterizes the desired prefix statistic itself. Thus, learning a function that satisfies Eq. (13) is equivalent to learning the targeted prefix-integral representation needed for interval statistics inference.

Training objective We parameterize the prefix statistic with a neural surrogate $\tilde{\boldsymbol{h}}_\theta(\mathbf{x}, s)$. The first-order identity formalized in Eq. (13) naturally yields a learning objective:

$$\mathcal{L}(\theta) = \mathbb{E}_{s \sim \mathrm{Unif}(0,1]} \left[\left\| \tilde{\boldsymbol{h}}_\theta(\mathbf{x}, s) - \mathrm{sg}\Big(\tilde{\boldsymbol{h}}_{\mathrm{tgt}}(\mathbf{x}, s) \Big) \right\|_2^2 \right], \text{ where } \tilde{\boldsymbol{h}}_{\mathrm{tgt}}(\mathbf{x}, s) = \boldsymbol{\psi}(\mathbf{x}, s) - s\, \partial_s \tilde{\boldsymbol{h}}_\theta(\mathbf{x}, s). \tag{14}$$

Here $\mathrm{sg}(\cdot)$ denotes stop-gradient. As discussed above, this formulation enforces the first-order identity while avoiding backpropagation through the derivative-dependent target. Note that each training sample requires only a local solver evaluation at $c = Q_\mu^{-1}(s)$. The network learns a global family of prefix statistics over quantile space, rather than separate predictors for separate intervals. Direct residual minimization is an alternative to stop-gradient training; Appendix B.5 compares their accuracy and training cost.

To relate the training objective to interval inference, define the cumulative prediction $\boldsymbol{A}_\theta(\mathbf{x}, s) = s \tilde{\boldsymbol{h}}_\theta(\mathbf{x}, s)$ and its local residual

$$\begin{aligned}\boldsymbol{\delta}_\theta(\mathbf{x}, s) &= \partial_s \boldsymbol{A}_\theta(\mathbf{x}, s) - \boldsymbol{\psi}(\mathbf{x}, s) \\ &= \tilde{\boldsymbol{h}}_\theta(\mathbf{x}, s) + s\, \partial_s \tilde{\boldsymbol{h}}_\theta(\mathbf{x}, s) - \boldsymbol{\psi}(\mathbf{x}, s).\end{aligned} \tag{15}$$

The following theorem expresses interval prediction error directly in terms of this residual.

Algorithm 1 Training pipeline of neural statistical functions $\tilde{\boldsymbol{h}}_\theta$.

Require: Dataset $\mathcal{D} = \{(\mathbf{x}, c, \mathbf{y})\}$, single-sample predictor $\boldsymbol{h}_{\theta_{\text{sample}}}$, finite-difference step Δs, data loss weight $\lambda_{\text{data}} = 0.1$

for each training step **do**

Sample a design $(\mathbf{x}, c, \mathbf{y}) \sim \mathcal{D}$

Sample $s_{\text{neural}} \sim \text{Unif}(0, 1]$ and set $\boldsymbol{\psi}_{\text{neural}} \leftarrow \boldsymbol{\psi}\big(\boldsymbol{h}_{\theta_{\text{sample}}}(\mathbf{x}, Q_\mu^{-1}(s_{\text{neural}}))\big)$ ▷ *Neural branch*

Set $s_{\text{data}} \leftarrow Q_\mu(c)$ and $\boldsymbol{\psi}_{\text{data}} \leftarrow \boldsymbol{\psi}(\mathbf{y})$ ▷ *Data branch*

Define $\tilde{\boldsymbol{h}}_{\text{tgt}}(\mathbf{x}, s, \boldsymbol{\psi}) = \boldsymbol{\psi} - s\,\partial_s \tilde{\boldsymbol{h}}_\theta(\mathbf{x}, s)$ with $\partial_s \tilde{\boldsymbol{h}}_\theta$ estimated by Δs-step finite differences

Compute neural-predictor-based loss

$$\mathcal{L}_{\text{neural}} = \big\|\tilde{\boldsymbol{h}}_\theta(\mathbf{x}, s_{\text{neural}}) - \text{sg}\big(\tilde{\boldsymbol{h}}_{\text{tgt}}(\mathbf{x}, s_{\text{neural}}, \boldsymbol{\psi}_{\text{neural}})\big)\big\|_2^2$$

Compute simulation-data-based loss

$$\mathcal{L}_{\text{data}} = \big\|\tilde{\boldsymbol{h}}_\theta(\mathbf{x}, s_{\text{data}}) - \text{sg}\big(\tilde{\boldsymbol{h}}_{\text{tgt}}(\mathbf{x}, s_{\text{data}}, \boldsymbol{\psi}_{\text{data}})\big)\big\|_2^2$$

Take a gradient step on $\mathcal{L} = \mathcal{L}_{\text{neural}} + \lambda_{\text{data}}\mathcal{L}_{\text{data}}$

end for

Theorem 3 (Exact residual identity and interval inference error)**.** *Fix* $\mathbf{x}$ *and suppress it in the notation. Suppose the cumulative prediction* $\boldsymbol{A}_\theta$ *is absolutely continuous on* $[0, 1]$*,* $\boldsymbol{A}_\theta(0) = 0$*, and* $\boldsymbol{\psi} \in L^1([0, 1])$*. Let* $\boldsymbol{\delta}_\theta$ *be the residual in Eq.* (15)*, defined a.e. For* $I = [s_0, s_1]$ *of mass* $m = s_1 - s_0 > 0$*, let* $\widehat{\boldsymbol{z}}_I = (\boldsymbol{A}_\theta(s_1) - \boldsymbol{A}_\theta(s_0))/m$ *and* $\boldsymbol{z}_I = m^{-1}\int_I \boldsymbol{\psi}(s)\,\mathrm{d}s$*. Then*

$$\widehat{\boldsymbol{z}}_I - \boldsymbol{z}_I = \frac{1}{m}\int_I \boldsymbol{\delta}_\theta(s)\,\mathrm{d}s, \tag{16}$$

$$\|\widehat{\boldsymbol{z}}_I - \boldsymbol{z}_I\| \le \frac{1}{m}\int_I \|\boldsymbol{\delta}_\theta(s)\|\,\mathrm{d}s. \tag{17}$$

Theorem 3 shows that prediction error in the transformed interval average equals the interval average of the local residual. Small residuals therefore control this error, while cancellation can yield accurate interval predictions even when local residuals are nonzero.

Proposition 4 (Narrow-interval sensitivity)**.** *Fix* $\mathbf{x}$ *and let* $\boldsymbol{\epsilon}_\theta(\mathbf{x}, s) := \tilde{\boldsymbol{h}}_\theta(\mathbf{x}, s) - \tilde{\boldsymbol{h}}(\mathbf{x}, s)$*, where* $\tilde{\boldsymbol{h}}$ *denotes the true prefix statistics. Fix* $s^\star \in (0, 1)$ *and suppose* $\boldsymbol{\epsilon}_\theta(\mathbf{x}, \cdot)$ *is continuously differentiable in a neighborhood of* $s^\star$*. Then*

$$\lim_{s_0, s_1 \to s^\star, s_1 > s_0} \frac{s_1\tilde{\boldsymbol{h}}_\theta(\mathbf{x}, s_1) - s_0\tilde{\boldsymbol{h}}_\theta(\mathbf{x}, s_0)}{s_1 - s_0} - \frac{s_1\tilde{\boldsymbol{h}}(\mathbf{x}, s_1) - s_0\tilde{\boldsymbol{h}}(\mathbf{x}, s_0)}{s_1 - s_0} = \boldsymbol{\epsilon}_\theta(\mathbf{x}, s^\star) + s^\star\partial_s\boldsymbol{\epsilon}_\theta(\mathbf{x}, s^\star). \tag{18}$$

Proposition 4 describes the fine-resolution limit: as intervals shrink, their averaged residual approaches the local residual. Since

$$\boldsymbol{\delta}_\theta(\mathbf{x}, s) = \boldsymbol{\epsilon}_\theta(\mathbf{x}, s) + s\,\partial_s\boldsymbol{\epsilon}_\theta(\mathbf{x}, s), \tag{19}$$

small prefix-value error alone does not ensure accurate narrow-interval queries. Thus, the residual underlying the training objective in Eq. (14) determines the limiting error in the transformed interval average.

Query resolution and representation. The residual identity also reveals how aggregation can reduce representation requirements: local errors that cancel within a query interval do not affect its transformed average. For interval means, we establish matching approximation bounds showing that sufficiently broad queries permit a better approximation rate than full-response reconstruction with the same number of summary coordinates. The advantage depends on query resolution and diminishes as intervals narrow. Appendix A.4 gives the precise result, construction, and matching lower bound.

Hybrid supervision $\tilde{\boldsymbol{h}}_{\text{tgt}}$ The gradient-matching objective in Eq. (14) only requires evaluating the locally transformed signal $\boldsymbol{\psi}(\mathbf{x}, s)$ at a single quantile point $s \in (0, 1]$ per training step. As demon-

strated in Algorithm 1, we adopt a hybrid scheme for practical training, which includes data and neural predictor branches. Specifically, the data supervision directly assigns the ground truth quantity $\mathbf{y}$ as the input to ψ and the neural predictor branch relies on a pretrained single-condition predictor to generate the corresponding estimate. The two regimes are complementary: the pretrained predictor provides dense, continuous coverage of s but inherits predictor bias, while data supervision is unbiased at the observed samples but restricted to the discrete c_i in the dataset. We therefore optimize both losses jointly, using the neural-predictor loss to provide continuous coverage and the data loss, weighted by λ_{data}, to anchor training to ground-truth simulations, i.e., $\mathcal{L} = \mathcal{L}_{\text{neural}} + \lambda_{\text{data}}\mathcal{L}_{\text{data}}$.

4 Experiments

Neural statistical functions support diverse statistics across physical systems, with overall performance summarized in Figure 2. We present three representative tasks in the main text (Table 2): mean energy in dynamical systems, pressure quantiles in aerodynamics, and maximum stress in crash processes. We compare NSF with Monte Carlo estimation across the three tasks. Full statistical results, additional baseline comparisons, and ablation studies are provided in Appendix B.

Table 2: Summary of main text experiments.

Systems	Targeted Statistics	Varied Condition
2D Dynamics	Mean energy	Time
Airplane Aero	Quantile pressure	Angle of attack
Car Crash	Maximum stress	Impact angle

Implementation details All experiments are conducted in PyTorch [25] on NVIDIA V100 32GB GPUs. We train neural statistical functions for 2,000 epochs with AdamW [20] and OneCycleLR [30]. They share the architecture of the corresponding single-condition inference models: an MLP for 2D dynamics [10] and Transolver [34] for aerodynamics and crash. MC experiments are repeated over 5 seeds; curves show the mean and shaded bands indicate one standard error.

Reference statistics. To evaluate model performance, we construct reference statistics using the following two approaches. *(i)* Dense inference with single-condition predictors. Specifically, we evaluate the single-condition model 1,000 times, with the condition c sampled from the considered interval. This provides a strong reference for statistical estimation based on single-condition models. *(ii)* Dense computation with numerical solvers. When available, we run numerical simulations [4, 7] under different conditions and compute statistics from the resulting dense sweeps. Specifically, for the 2D dynamics task, we directly compute the analytic statistics from classical mechanics. For the car crash task, we simulate 280 samples with uniformly sampled impact angles and compute statistics. For the aerodynamics task [7], since the simulation configuration is not released, we only use approach (i) as the reference. Approach (ii) requires hundreds of numerical simulations for each design, which is computationally prohibitive in practical deployment.

4.1 Mean of dynamical systems

Setup We evaluate interval-conditioned mean energy on a synthetic 2D dynamical system with normalized time $s \in [0, 1]$. Each trajectory (Figure 3) has 11 parameters controlling spiral motion, a 40-cycle angular component, and localized radial and angular transients. These produce sharp variations in instantaneous energy. We use 3,000 training and 900 held-out test designs. Writing the trajectory in polar coordinates, the target is

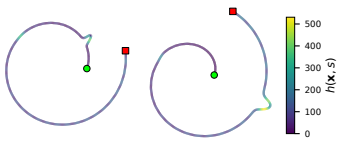

Figure 3: Example trajectories.

$$\mathbf{y}(s) = r(s)\big(\cos\eta(s), \sin\eta(s)\big),$$
$$h_{\text{mean}}(\mathbf{x}, [s_0, s_1]) = \frac{1}{s_1 - s_0}\int_{s_0}^{s_1}\left[r'(s)^2 + \big(r(s)\eta'(s)\big)^2\right] \mathrm{d}s.$$

We compare against IID MC with $K = 50, 100, 200, 400$ and 40-evaluation midpoint, Gauss–Legendre, scrambled Sobol, and stratified rules. All baselines use the exact instantaneous response. References are an independent 1,000-sample MC mean and the analytic interval mean. NSF uses a four-hidden-layer SiLU MLP [26, 28] of width 512, trained with stop gradients.

ψ implementation Here $c = s$ is uniform, so Q_μ^{-1} is the identity and $\psi(\mathbf{x}, s) = h(\mathbf{x}, s)$ for the mean. We evaluate eight equally spaced interval widths $s_1 - s_0 \in [0.90, 1.00]$.

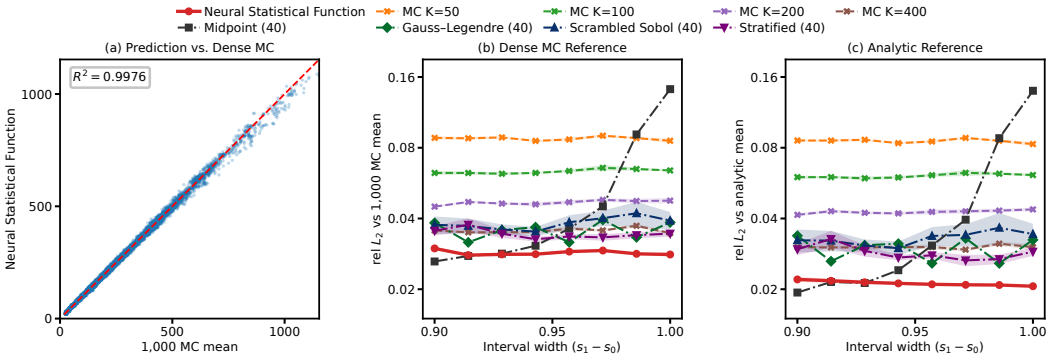

Figure 4: Mean estimation on the 2D dynamical system: (a) NSF predictions versus an independent 1,000-sample MC reference; relative ℓ_2 error against (b) the MC reference and (c) the analytic interval mean. Shaded bands indicate one standard error.

Results With only two endpoint evaluations, NSF outperforms MC with up to $K = 400$ at every tested width and achieves lower average error than the four 40-evaluation numerical baselines. Its advantage is strongest on wider intervals. The similar trends obtained when evaluating against the dense 1,000-sample Monte Carlo reference and the analytic interval mean in Figures 4(b,c) further indicate that the dense Monte Carlo reference is sufficiently accurate for practical comparison. Additional diagnostics appear in Appendix B.5.

4.2 α-quantile of aerodynamic systems

Setup We evaluate interval-conditioned quantile prediction on the NASA-CRM [7], which provides high-quality simulations for flying airplanes. This benchmark contains 149 samples, where each sample is simulated under various angles of attack (AoA) and geometries. The quantity of interest $\boldsymbol{h}(\mathbf{x}, c)$ is the point-wise pressure coefficient on the airplane surface $\mathbf{x}$. The uncertain parameter c is AoA over $[c_{\min}, c_{\max}] = [-2.5°, 7.5°]$, treated as uniform, and queries are defined over intervals $[c_0, c_1]$ of varying width. We adopt 105 samples for training and the remaining 44 samples with varied geometries for testing. The targeted statistic is the conditional α-quantile (Eq. (8)) for each geometry.

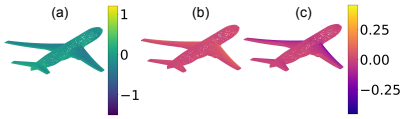

Figure 5: 0.9-quantile of point-wise pressure over a 9.8° AoA interval: (a) dense-MC reference, (b) neural statistical function error, and (c) MC ($K = 10$) error.

As discussed above, since we cannot access the simulation configuration of this dataset, we adopt a pre-trained Transolver model [34] with dense inference to approximate the ground truth statistics. Here Transolver is an established neural simulator, which can produce high-accuracy point-wise simulation. We use the same architecture in training neural statistical function $\tilde{\boldsymbol{h}}_\theta$.

ψ implementation As shown in Eq. (9)-(10), ψ is defined as an indicator function, which is unstable in calculating differentiation. Thus, we smooth ψ with a sigmoid-style approximation:

$$\psi(\mathbf{x}, s; \mathbf{y}) = \big(1 + \exp\big(-\frac{\mathbf{y} - \boldsymbol{h}(\mathbf{x}, Q_\mu^{-1}(s))}{\varepsilon}\big)\big)^{-1} \approx \mathbb{1}\big\{\boldsymbol{h}(\mathbf{x}, Q_\mu^{-1}(s)) \le \mathbf{y}\big\}, \tag{20}$$

and the smoothing width ε is annealed from 0.1 to 0.01 during training, in units of one standard deviation of the normalized surface pressure response. In practice, we use five binary-search steps per interval query, requiring ten logical endpoint evaluations.

Results As shown in Figure 6, neural statistical functions consistently outperform MC with $K = 10$, with larger gains on wider AoA intervals that encompass a broader range of aerodynamic

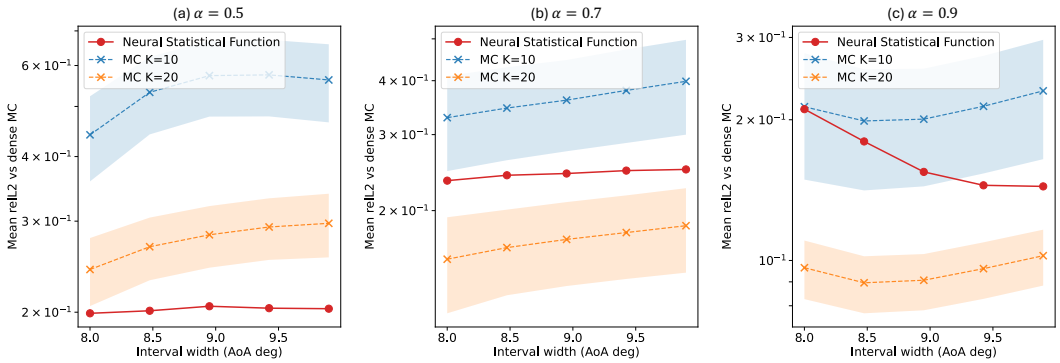

Figure 6: Width-binned relative ℓ_2 error for interval-conditioned pressure quantile prediction on NASA-CRM: (a) $\alpha = 0.5$, (b) $\alpha = 0.7$, and (c) $\alpha = 0.9$. Errors are computed against the dense Transolver reference; red denotes neural statistical function and dashed curves denote MC baselines.

responses. Quantile prediction is more demanding than mean estimation because it requires learning and inverting an interval-conditioned CDF. NSF nevertheless improves accuracy at the same evaluation budget, demonstrating its applicability beyond mean prediction. MC with $K = 20$ benefits from additional samples and remains competitive at some quantile levels, while NSF requires half as many evaluations.

Additionally, we also provide error visualization in Figure 5, where our model is more accurate and spatially coherent than the MC ($K = 10$) error map, especially around complex wing regions where the pressure field changes rapidly. This suggests that neural statistical functions capture structured spatial variation in the conditional quantile more reliably than sparse Monte Carlo sampling.

4.3 Maximum of crash processes

Setup We evaluate the maximum estimate conditioned on intervals in the Car-Crash simulated with OpenRadioss [3]. This benchmark contains 280 cases for the industrial-standard National Crash Analysis Center Neon model, where each sample is tested with different impact angles and will present different geometry deformations during crash. Here, the quantity of interest $\boldsymbol{h}(\mathbf{x}, c)$ is the element-wise maximum 2D von Mises stress during the crash process and $\mathbf{x}$ denotes the geometry position. The uncertain parameter c is the impact angle over $[c_{\min}, c_{\max}] = [-55°, 55°]$, treated as uniform, and queries are defined over intervals $[c_0, c_1]$ of varying width. We adopt 80% of the samples (224 cases) for training and the remaining 20% (56 cases) for testing. The targeted statistic is element-wise maximum stress under different impact angle intervals $[c_0, c_1]$.

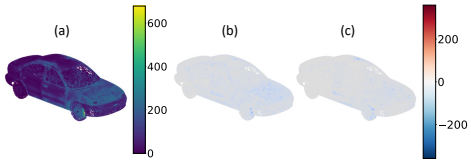

Figure 7: Example maximum stress field over a 100° impact angle interval: (a) dense Transolver reference, (b) neural statistical function error, and (c) MC $K = 10$ error, both relative to the dense Transolver reference.

As discussed before, we adopt the two approaches to calculate the reference statistics. For dense inference of single-sample predictors, we adopt dense MC evaluation of pre-trained Transolver [34] to generate a reference, where the same architecture is adopted in training neural statistical functions $\tilde{\boldsymbol{h}}_\theta$. Another reference is calculated from all 280 numerical simulations for dense evaluation.

ψ implementation As formalized in Eq. (7), we adopt a log-sum-exp formalization [24] to approximate the maximum function, which introduces β as a smoothing temperature. Specifically, we set $\beta = 10$ based on training loss curves to balance training stability and approximation accuracy.

Results As shown in Figure 8(a), neural statistical functions achieve accuracy competitive with or better than MC with $K = 10$ on wider impact-angle intervals, using only two endpoint evaluations of

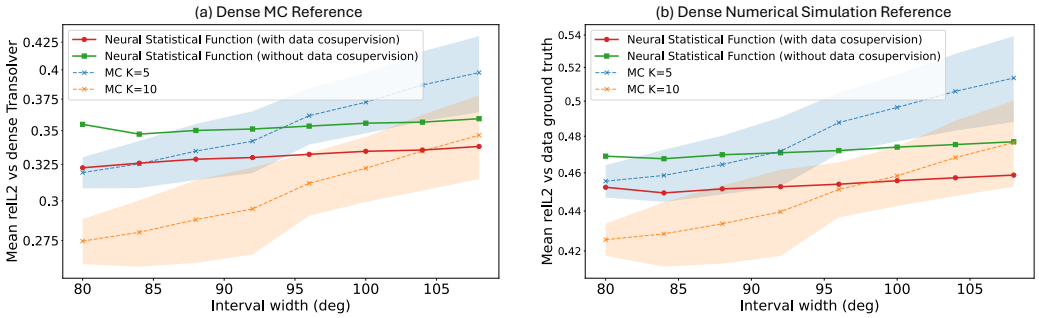

Figure 8: Interval-conditioned maximum stress prediction on the Car-Crash. Width-binned relative ℓ_2 error is reported for neural statistical functions and MC with $K \in \{5, 10\}$. (a) Errors against the dense Transolver reference. (b) Errors against the dense numerical-simulation reference.

$\tilde{\boldsymbol{h}}_\theta$. Wider intervals encompass a broader range of nonlinear crash responses, making it challenging for sparse Monte Carlo samples to capture peak stress.

Effect of hybrid supervision As an ablation, we further examine the role of data supervision in model training in Figure 8(b). Under the ground truth data reference, the neural statistical function with data supervision improves over pure neural predictor supervision, indicating that the additional simulation anchors help align the learned statistic with the real stress envelope observed in the dataset. Neural-only training remains stable and competitive, while hybrid supervision improves agreement with the simulation reference. Appendix B.6 further examines sensitivity to imperfect teachers.

5 Conclusion and discussion

We introduced neural statistical functions, a new family of neural networks that directly infer statistical quantities over continuous operating conditions, reducing the inference cost of stochastic analysis. Building on the prefix perspective, we developed a unified framework for diverse statistical functions and derived an identity-based learning objective for efficient training. Our analysis identifies residual cancellation as a mechanism for accurate interval inference and establishes a sharp resolution-dependent representation advantage for means. Neural statistical functions accelerate statistical estimation for complex physical processes with up to $200\times$ lower inference latency, demonstrating strong potential to advance scientific discovery and engineering design (see the example in Appendix B.3).

Limitations Our experiments cover one- and two-dimensional condition domains. Rectangular queries require $2^{d_{\text{cond}}}$ prefix evaluations, limiting scalability as the number of condition variables grows; dependent condition distributions and non-rectangular regions require further development (Appendix B.2). Thus, designing sampling-free, low-latency neural statistical functions for extremely high-dimensional condition spaces remains an important open challenge.

Acknowledgments

This research received no external funding or third-party financial support. The authors declare no financial or non-financial competing interests.

References

[1] Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023.

[2] 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. *TMLR*, 2025.

[3] Altair Engineering Inc. Altair physicsai. `https://www.altair.com/physicsai`, 2026. Accessed: 2026-01-06.

[4] Altair Engineering Inc. Altair radioss. `https://www.openradioss.org`, 2026. Accessed: 2026-01-06.

[5] Ansys Inc. Ansys simai. `https://www.ansys.com/products/simai`, 2026. Accessed: 2026-01-06.

[6] Kamyar Azizzadenesheli, Nikola Kovachki, Zongyi Li, Miguel Liu-Schiaffini, Jean Kossaifi, and Anima Anandkumar. Neural operators for accelerating scientific simulations and design. *Nature Reviews Physics*, 2024.

[7] Philipp Bekemeyer, Nathan Hariharan, Andrew M Wissink, and Jason Cornelius. Introduction of applied aerodynamics surrogate modeling benchmark cases. In *AIAA SCITECH 2025 Forum*, 2025.

[8] Kaifeng Bi, Lingxi Xie, Hengheng Zhang, Xin Chen, Xiaotao Gu, and Qi Tian. Accurate medium-range global weather forecasting with 3d neural networks. *Nature*, 2023.

[9] Joël Brézillon and Mohammad Abu-Zurayk. Aerodynamic inverse design framework using discrete adjoint method. In Andreas Dillmann, Gerd Heller, Hans-Peter Kreplin, Wolfgang Nitsche, and Inken Peltzer, editors, *New Results in Numerical and Experimental Fluid Mechanics VIII*, volume 121 of *Notes on Numerical Fluid Mechanics and Multidisciplinary Design*, pages 489–496. Springer, 2013. doi: 10.1007/978-3-642-35680-3_58.

[10] Ricky TQ Chen, Yulia Rubanova, Jesse Bettencourt, and David K Duvenaud. Neural ordinary differential equations. *NeurIPS*, 2018.

[11] Emilien Dupont, Arnaud Doucet, and Yee Whye Teh. Augmented neural odes. *NeurIPS*, 2019.

[12] Yarin Gal and Zoubin Ghahramani. Dropout as a bayesian approximation: Representing model uncertainty in deep learning. In *ICML*, 2016.

[13] Zhengyang Geng, Mingyang Deng, Xingjian Bai, J Zico Kolter, and Kaiming He. Mean flows for one-step generative modeling. *NeurIPS*, 2025.

[14] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *NeurIPS*, 2020.

[15] John Jumper, Richard Evans, Alexander Pritzel, Tim Green, Michael Figurnov, Olaf Ronneberger, Kathryn Tunyasuvunakool, Russ Bates, Augustin Žídek, Anna Potapenko, et al. Highly accurate protein structure prediction with alphafold. *Nature*, 2021.

[16] Alex Kendall, Yarin Gal, and Roberto Cipolla. Multi-task learning using uncertainty to weigh losses for scene geometry and semantics. In *CVPR*, 2018.

[17] Nikola Kovachki, Zongyi Li, Burigede Liu, Kamyar Azizzadenesheli, Kaushik Bhattacharya, Andrew Stuart, and Anima Anandkumar. Neural operator: Learning maps between function spaces with applications to pdes. *JMLR*, 2023.

[18] Timothy Langlois, Ariel Shamir, Daniel Dror, Wojciech Matusik, and David IW Levin. Stochastic structural analysis for context-aware design and fabrication. *ACM Transactions on Graphics (TOG)*, 2016.

[19] 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 *ICLR*, 2021.

[20] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *ICLR*, 2019.

[21] Cheng Lu, Yuhao Zhou, Fan Bao, Jianfei Chen, Chongxuan Li, and Jun Zhu. Dpm-solver: A fast ode solver for diffusion probabilistic model sampling in around 10 steps. *NeurIPS*, 2022.

[22] 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 *ICML*, 2025.

[23] Mohammad Amin Nabian, Sudeep Chavare, Deepak Akhare, Rishikesh Ranade, Ram Cherukuri, and Srinivas Tadepalli. Automotive crash dynamics modeling accelerated with machine learning. *arXiv preprint arXiv:2510.15201*, 2025.

[24] Yurii Nesterov. Smooth minimization of non-smooth functions. *Mathematical Programming*, 2005.

[25] Adam Paszke, S. Gross, Francisco Massa, A. Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Z. Lin, N. Gimelshein, L. Antiga, Alban Desmaison, Andreas Köpf, Edward Yang, Zach DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. In *NeurIPS*, 2019.

[26] Prajit Ramachandran, Barret Zoph, and Quoc V Le. Searching for activation functions. *arXiv preprint arXiv:1710.05941*, 2017.

[27] Yulia Rubanova, Ricky TQ Chen, and David K Duvenaud. Latent ordinary differential equations for irregularly-sampled time series. *NeurIPS*, 2019.

[28] Noam Shazeer. Glu variants improve transformer. *arXiv preprint arXiv:2002.05202*, 2020.

[29] Jonathan W. Siegel. Sharp lower bounds on the manifold widths of Sobolev and Besov spaces. *arXiv preprint arXiv:2402.04407*, 2024. URL `https://arxiv.org/abs/2402.04407`.

[30] Leslie N Smith and Nicholay Topin. Super-convergence: Very fast training of neural networks using large learning rates. In *Artificial intelligence and machine learning for multi-domain operations applications*. SPIE, 2019.

[31] Yang Song, Prafulla Dhariwal, Mark Chen, and Ilya Sutskever. Consistency models. *ICML*, 2023.

[32] George Stefanou. The stochastic finite element method: past, present and future. *CMAME*, 2009.

[33] Hanchen Wang, Tianfan Fu, Yuanqi Du, Wenhao Gao, Kexin Huang, Ziming Liu, Payal Chandak, Shengchao Liu, Peter Van Katwyk, Andreea Deac, et al. Scientific discovery in the age of artificial intelligence. *Nature*, 2023.

[34] Haixu Wu, Huakun Luo, Haowen Wang, Jianmin Wang, and Mingsheng Long. Transolver: A fast transformer solver for pdes on general geometries. In *ICML*, 2024.

[35] Haixu Wu, Minghao Guo, Zongyi Li, Zhiyang Dou, Mingsheng Long, Kaiming He, and Wojciech Matusik. Geopt: Scaling physics simulation via lifted geometric pre-training. In *ICML*, 2026.

[36] 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. *arXiv preprint arXiv:2602.04940*, 2026.

A Theoretical analysis and proofs

This section provides proofs for the main-text results and develops the resolution-dependent mean-compression analysis.

A.1 Proof of Proposition 2

Proof. We fix $\mathbf{x}$ throughout the proof and omit it from the notation whenever no confusion arises.

We first show that the prefix-integral representation implies the first-order identity. Suppose that

$$\tilde{\boldsymbol{h}}(\mathbf{x}, s) = \frac{1}{s} \int_0^s \boldsymbol{\psi}(\mathbf{x}, t)\, \mathrm{d}t, \qquad s \in (0, 1].$$

Then

$$s\, \tilde{\boldsymbol{h}}(\mathbf{x}, s) = \int_0^s \boldsymbol{\psi}(\mathbf{x}, t)\, \mathrm{d}t.$$

By the fundamental theorem of calculus,

$$\frac{\mathrm{d}}{\mathrm{d}s}\Big[s\, \tilde{\boldsymbol{h}}(\mathbf{x}, s)\Big] = \boldsymbol{\psi}(\mathbf{x}, s) \qquad \text{for a.e. } s \in (0, 1].$$

Using the product rule, we obtain

$$s\, \partial_s \tilde{\boldsymbol{h}}(\mathbf{x}, s) + \tilde{\boldsymbol{h}}(\mathbf{x}, s) = \boldsymbol{\psi}(\mathbf{x}, s),$$

which is precisely Eq. (13). Moreover,

$$\lim_{s \to 0} s\, \tilde{\boldsymbol{h}}(\mathbf{x}, s) = \lim_{s \to 0} \int_0^s \boldsymbol{\psi}(\mathbf{x}, t)\, \mathrm{d}t = 0,$$

where the last equality follows from $\boldsymbol{\psi}(\mathbf{x}, \cdot) \in L^1(0, 1]$.

Conversely, suppose that $\tilde{\boldsymbol{h}}$ satisfies Eq. (13) on $(0, 1]$ together with the boundary condition

$$\lim_{s \to 0} s\, \tilde{\boldsymbol{h}}(\mathbf{x}, s) = 0.$$

By the product rule,

$$\frac{\mathrm{d}}{\mathrm{d}s}\Big[s\, \tilde{\boldsymbol{h}}(\mathbf{x}, s)\Big] = s\, \partial_s \tilde{\boldsymbol{h}}(\mathbf{x}, s) + \tilde{\boldsymbol{h}}(\mathbf{x}, s).$$

Using Eq. (13), this gives

$$\frac{\mathrm{d}}{\mathrm{d}s}\Big[s\, \tilde{\boldsymbol{h}}(\mathbf{x}, s)\Big] = \boldsymbol{\psi}(\mathbf{x}, s).$$

Integrating both sides from u to s, where $0 < u < s \leq 1$, yields

$$s\, \tilde{\boldsymbol{h}}(\mathbf{x}, s) - u\, \tilde{\boldsymbol{h}}(\mathbf{x}, u) = \int_u^s \boldsymbol{\psi}(\mathbf{x}, t)\, \mathrm{d}t.$$

Taking the limit $u \to 0$ and using the boundary condition, we obtain

$$s\, \tilde{\boldsymbol{h}}(\mathbf{x}, s) = \int_0^s \boldsymbol{\psi}(\mathbf{x}, t)\, \mathrm{d}t.$$

Dividing both sides by s proves

$$\tilde{\boldsymbol{h}}(\mathbf{x}, s) = \frac{1}{s} \int_0^s \boldsymbol{\psi}(\mathbf{x}, t)\, \mathrm{d}t,$$

which is Eq. (5). This completes the proof. □

A.2 Proof of Theorem 3

Proof. Fix $\mathbf{x}$ and suppress it in the notation. Absolute continuity gives

$$\boldsymbol{A}_\theta(s_1) - \boldsymbol{A}_\theta(s_0) - \int_{s_0}^{s_1} \boldsymbol{\psi}(s)\, \mathrm{d}s = \int_{s_0}^{s_1} (\boldsymbol{A}'_\theta(s) - \boldsymbol{\psi}(s))\, \mathrm{d}s = \int_{s_0}^{s_1} \boldsymbol{\delta}_\theta(s)\, \mathrm{d}s.$$

Dividing by $m = s_1 - s_0$ proves Eq. (16). Taking norms and applying the triangle inequality proves Eq. (17). For $s_0 = 0$, the endpoint term is defined by $\boldsymbol{A}_\theta(0) = 0$. □

A.3 Proof of Proposition 4

Proof. Fix $\mathbf{x}$ and omit it from the notation when no confusion arises. By definition,

$$\boldsymbol{\epsilon}_\theta(\mathbf{x}, s) = \tilde{\boldsymbol{h}}_\theta(\mathbf{x}, s) - \tilde{\boldsymbol{h}}(\mathbf{x}, s).$$

Therefore,

$$\frac{s_1\tilde{\boldsymbol{h}}_\theta(\mathbf{x}, s_1) - s_0\tilde{\boldsymbol{h}}_\theta(\mathbf{x}, s_0)}{s_1 - s_0} - \frac{s_1\tilde{\boldsymbol{h}}(\mathbf{x}, s_1) - s_0\tilde{\boldsymbol{h}}(\mathbf{x}, s_0)}{s_1 - s_0} = \frac{s_1\boldsymbol{\epsilon}_\theta(\mathbf{x}, s_1) - s_0\boldsymbol{\epsilon}_\theta(\mathbf{x}, s_0)}{s_1 - s_0}.$$

Define

$$\boldsymbol{g}_\theta(\mathbf{x}, s) := s\,\boldsymbol{\epsilon}_\theta(\mathbf{x}, s).$$

Since $\boldsymbol{\epsilon}_\theta(\mathbf{x}, \cdot)$ is continuously differentiable in a neighborhood of $s^\star$, so is $\boldsymbol{g}_\theta(\mathbf{x}, \cdot)$. For s_0, s_1 in this neighborhood, the fundamental theorem of calculus gives

$$\frac{\boldsymbol{g}_\theta(\mathbf{x}, s_1) - \boldsymbol{g}_\theta(\mathbf{x}, s_0)}{s_1 - s_0} = \frac{1}{s_1 - s_0}\int_{s_0}^{s_1} \partial_s\boldsymbol{g}_\theta(\mathbf{x}, t)\,\mathrm{d}t.$$

By continuity of $\partial_s\boldsymbol{g}_\theta$ at $s^\star$, taking the limit as $s_0, s_1 \to s^\star$ with $s_1 > s_0$ yields

$$\lim_{\substack{s_0, s_1 \to s^\star \\ s_1 > s_0}} \frac{\boldsymbol{g}_\theta(\mathbf{x}, s_1) - \boldsymbol{g}_\theta(\mathbf{x}, s_0)}{s_1 - s_0} = \partial_s\boldsymbol{g}_\theta(\mathbf{x}, s^\star).$$

Using the product rule,

$$\partial_s\boldsymbol{g}_\theta(\mathbf{x}, s^\star) = \partial_s\Big[s\,\boldsymbol{\epsilon}_\theta(\mathbf{x}, s)\Big]_{s=s^\star} = \boldsymbol{\epsilon}_\theta(\mathbf{x}, s^\star) + s^\star\partial_s\boldsymbol{\epsilon}_\theta(\mathbf{x}, s^\star).$$

Combining the above identities gives

$$\lim_{s_0, s_1 \to s^\star,\, s_1 > s_0} \frac{s_1\tilde{\boldsymbol{h}}_\theta(\mathbf{x}, s_1) - s_0\tilde{\boldsymbol{h}}_\theta(\mathbf{x}, s_0)}{s_1 - s_0} - \frac{s_1\tilde{\boldsymbol{h}}(\mathbf{x}, s_1) - s_0\tilde{\boldsymbol{h}}(\mathbf{x}, s_0)}{s_1 - s_0} = \boldsymbol{\epsilon}_\theta(\mathbf{x}, s^\star) + s^\star\partial_s\boldsymbol{\epsilon}_\theta(\mathbf{x}, s^\star),$$

which proves Eq. (18). □

A.4 Resolution-dependent mean compression

Interval averaging permits cancellation of local errors, allowing accurate mean prediction without equally accurate reconstruction of the underlying response function. We characterize this representational advantage as a function of query resolution for a fixed geometry and a scalar response f in quantile space. For $0 < \nu \le 1$, let

$$\mathcal{F}_\nu = \{f \in C([0,1]) : \|f\|_\infty \le 1,\ |f(s) - f(t)| \le |s - t|^\nu \text{ for all } s, t \in [0,1]\}.$$

Write $Q_I f = |I|^{-1}\int_I f(s)\,\mathrm{d}s$. Let $e_n^{\mathrm{mean}}(m_0)$ be the smallest worst-case absolute error over $f \in \mathcal{F}_\nu$ and all intervals $|I| \ge m_0$, using an encoding $E : \mathcal{F}_\nu \to \mathbb{R}^n$ continuous in the uniform norm and an arbitrary decoder given $E(f)$ and I. Here n is the number of coordinates in the encoded representation.

Theorem 5 (Sharp resolution-dependent mean compression)**.** *For $n \ge 2$ and $0 < m_0 \le 1/2$,*

$$e_n^{\mathrm{mean}}(m_0) \asymp_\nu n^{-\nu}\min\{1, (nm_0)^{-1}\}. \tag{21}$$

Here the comparison constants depend only on ν. On the same class, optimal full-response reconstruction in uniform norm from a continuous n-coordinate encoding has error of order $n^{-\nu}$. The mean upper bound is attained by a piecewise-linear prefix potential whose slopes are the response averages on n equal condition cells.

Cancellation mechanism. Let $P_n f$ equal the response average in each cell of width $\Delta = 1/n$, and set $A_n(s) = \int_0^s P_n f(t)\,\mathrm{d}t$. Its residual $r = P_n f - f$ integrates to zero on every complete cell. By Theorem 3, only the two partial boundary cells contribute query error, yielding

$$|Q_I r| \le \min\{\Delta^\nu, \Delta^{\nu+1}/|I|\}.$$

For queries wider than the representation scale $1/n$, this gives an extra factor $1/(nm_0)$. For Lipschitz responses and fixed m_0, mean error is of order n^{-2}, compared with n^{-1} for response reconstruction; the gain disappears when queries resolve individual cells.

Proof setup. Let $\mathcal{I}_{m_0} = \{[a,b] \subset [0,1] : b - a \geq m_0\}$. More precisely,

$$e_n^{\text{mean}}(m_0) = \inf_{E,D} \sup_{f \in \mathcal{F}_\nu} \sup_{I \in \mathcal{I}_{m_0}} |D(E(f), I) - Q_I f|,$$

where $E : \mathcal{F}_\nu \to \mathbb{R}^n$ is continuous in the uniform norm and $D : \mathbb{R}^n \times \mathcal{I}_{m_0} \to \mathbb{R}$ is unrestricted. For comparison, define

$$e_n^{\text{response}} = \inf_{E,R} \sup_{f \in \mathcal{F}_\nu} \|R(E(f)) - f\|_\infty,$$

with the same encoder restriction and arbitrary function-valued decoder R.

Proof. Construction and upper bound. Partition $[0,1]$ into n equal cells C_j of length $\Delta = 1/n$. Encode f by the n cell averages $a_j = \Delta^{-1} \int_{C_j} f(s)\,\mathrm{d}s$, which are continuous linear functionals of f. Set $P_n f = a_j$ on C_j and

$$A_n(s) = \int_0^s P_n f(t)\,\mathrm{d}t, \qquad H_n(s) = A_n(s)/s \quad (s > 0).$$

Then A_n is an absolutely continuous, piecewise-linear prefix potential with $A_n(0) = 0$. The normalized prefix H_n extends continuously to zero with value a_1. Its endpoint differences return $Q_I P_n f$.

For $r = P_n f - f$, Hölder regularity gives $\|r\|_\infty \leq \Delta^\nu$, while $\int_{C_j} r = 0$ on each complete cell. Hence $B(s) = \int_0^s r(t)\,\mathrm{d}t$ only contains the last partial cell's residual. If that partial cell has length t, its complementary part has length $\Delta - t$ and opposite residual integral. Thus

$$|B(s)| \leq \Delta^\nu \min\{t, \Delta - t\} \leq \tfrac{1}{2}\Delta^{\nu+1}.$$

For $I = [a,b]$ of mass m, the exact residual identity now gives

$$|Q_I P_n f - Q_I f| = |Q_I r| = \frac{|B(b) - B(a)|}{m} \leq \min\{\Delta^\nu, \Delta^{\nu+1}/m\}.$$

Taking the supremum over $m \geq m_0$ proves the mean upper bound.

Matching lower bound. We use the classical Borsuk–Ulam theorem, as in nonlinear-width lower bounds [29]: a continuous map from the boundary of the unit cube in $\mathbb{R}^{n+1}$ to $\mathbb{R}^n$ identifies an antipodal pair. The cube boundary is antipodally homeomorphic to the n-sphere. Partition $[1/2, 1]$ into $n+1$ cells J_j of length $\ell = 1/[2(n+1)]$, and define disjoint bumps

$$b_j(s) = \begin{cases} 2^{\nu-1} \operatorname{dist}(s, \partial J_j)^\nu, & s \in J_j, \\ 0, & s \notin J_j. \end{cases} \qquad \|b_j\|_\infty = \frac{\ell^\nu}{2}, \quad \int b_j = \frac{\ell^{\nu+1}}{2(\nu+1)}.$$

For $\|u\|_\infty \leq 1$, let $f_u = \sum_{j=1}^{n+1} u_j b_j$. These responses belong to $\mathcal{F}_\nu$. Within a cell this follows from $|x^\nu - y^\nu| \leq |x - y|^\nu$ and the Lipschitz property of distance. Across distinct cells, if x, y are the distances to their respective cell boundaries, then $x + y \leq |s - t|$ and $2^{\nu-1}(x^\nu + y^\nu) \leq (x+y)^\nu$. The same argument with one distance zero covers points outside the bump support. The amplitude bound follows from disjoint supports and the stated bump height.

For any continuous encoder E, Borsuk–Ulam supplies u with $\|u\|_\infty = 1$ and $E(f_u) = E(f_{-u})$. Choose a cell $J_j = [c, c+\ell]$ with $|u_j| = 1$. Both $I_1 = [c - m_0, c]$ and $I_2 = [c - m_0, c + \ell]$ belong to $\mathcal{I}_{m_0}$, since $c \geq 1/2 \geq m_0$. Writing $\varepsilon_u = \sup_{I \in \mathcal{I}_{m_0}} |Q_I f_u|$, we obtain

$$\frac{\ell^{\nu+1}}{2(\nu+1)} = \left| \int_c^{c+\ell} f_u \right| = |(m_0 + \ell) Q_{I_2} f_u - m_0 Q_{I_1} f_u| \leq (2m_0 + \ell)\varepsilon_u.$$

The responses f_u and $-f_u$ receive the same code and hence the same decoded answer for each interval. Since their true answers have opposite signs, the worst-case error of any decoder is at least ε_u. Consequently

$$e_n^{\text{mean}}(m_0) \geq \frac{\ell^\nu}{2(\nu+1)(1 + 2m_0/\ell)} \gtrsim_\nu n^{-\nu} \min\{1, (nm_0)^{-1}\}.$$

Together with the upper bound, this proves Eq. (21).

Full-response comparison. The same indistinguishable pair satisfies $\|f_u - (-f_u)\|_\infty = \ell^\nu$, so any response decoder has worst-case uniform error at least $\ell^\nu/2 \gtrsim_\nu n^{-\nu}$. Conversely, encode the values of f at n equally spaced points including zero and one, and decode by piecewise-linear interpolation. On each grid interval, the interpolation error is at most $(n-1)^{-\nu}$ by Hölder regularity. This proves $e_n^{\text{response}} \asymp_\nu n^{-\nu}$. □

B Additional experiments and analysis

This section presents functional-output and joint-condition comparisons, additional benchmark statistics, model diagnostics and ablations, and teacher-robustness experiments, followed by computational cost and deployment analysis and numerical values for the main-text figures.

B.1 Interval means and functional-output prediction

Setup We further evaluate interval-mean pressure prediction on NASA-CRM, using the Transolver backbone to compare prefix statistics with prediction of the full condition-response curve. The functional-output baseline jointly predicts pressure fields at $M \in \{10, 20, 50\}$ uniformly spaced AoA values. For off-grid interval endpoints, we linearly interpolate this curve and exactly integrate the resulting piecewise-linear response. Errors are averaged over five AoA interval widths against the dense teacher reference. NSF uses stop-gradient training and two endpoint fields per query.

Table 3: NASA interval-mean prediction versus functional-output Transolver. NSF directly predicts the queried statistic with two endpoint fields. Storage counts FP32 output values at $N = 32{,}000$ nodes.

Method	Output fields	Mean relative ℓ_2	FP32 output storage
Functional output ($M = 10$)	10	0.08851	1.22 MiB
Functional output ($M = 20$)	20	0.06985	2.44 MiB
Functional output ($M = 50$)	50	0.05929	6.10 MiB
NSF, stop-gradient	2	0.05738	0.24 MiB

Results Stop-gradient NSF achieves lower error than the functional-output models at $M = 10, 20, 50$, while producing only two endpoint fields (Table 3). Its per-query FP32 output storage is $5\times$–$25\times$ smaller across these resolutions. Theorem 5 provides a complementary rationale: accurate interval means can require less information than uniformly accurate response reconstruction. This experiment compares the particular trained models; the theorem does not imply that every functional-output estimator is suboptimal for mean queries. In particular, a representation retaining cell averages can attain the same constructive query rate. Evaluating the NSF prefix at the requested endpoints also avoids interpolation from a fixed condition grid.

B.2 Multidimensional extension and joint AoA–Mach evaluation

Formulation For independent condition variables, marginal quantile transforms give jointly uniform coordinates in $[0, 1]^{d_{\mathrm{cond}}}$. The multidimensional prefix integrates the transformed local response over the rectangle from the origin to the queried condition coordinates. In two dimensions, the unnormalized prefix $G_\theta(\mathbf{x}, s_1, s_2) = s_1 s_2 \tilde{\boldsymbol{h}}_\theta(\mathbf{x}, s_1, s_2)$ approximates the cumulative response over $[0, s_1] \times [0, s_2]$. For a uniformly weighted query rectangle $R = [a_1, b_1] \times [a_2, b_2]$, its mean is recovered by four-corner inclusion–exclusion:

$$\widehat{\boldsymbol{h}}_R(\mathbf{x}) = \frac{G_\theta(\mathbf{x}, b_1, b_2) - G_\theta(\mathbf{x}, a_1, b_2) - G_\theta(\mathbf{x}, b_1, a_2) + G_\theta(\mathbf{x}, a_1, a_2)}{(b_1 - a_1)(b_2 - a_2)}. \tag{22}$$

Setup We evaluate NASA mean prediction under joint AoA–Mach conditions, normalizing the two variables to (s_1, s_2). Queries use rectangles with an AoA width of 9.5° and a Mach width of 0.35; MC samples condition pairs uniformly from the same rectangles.

Table 4: Joint AoA–Mach interval-mean prediction on NASA. Four-corner NSF achieves lower error than MC with up to 64 model evaluations.

Method	Logical model evaluations	Mean relative ℓ_2 error
2D NSF	**4**	**0.05519**
MC	4	0.27790
MC	16	0.14406
MC	32	0.10300
MC	64	0.07203

Results At the matched four-evaluation budget, NSF reduces error by approximately 80% relative to MC (Table 4). It also achieves approximately 23% lower error than MC-64 while using $16\times$ fewer evaluations. This establishes the practicality of prefix statistics for low-dimensional, axis-aligned box queries, beyond the scalar-condition setting.

B.3 Aerodynamic configuration selection under uncertainty

Setup Matching a prescribed pressure distribution is an established objective in aerodynamic inverse design [9]. Motivated by this objective, we evaluate neural statistical functions on a configuration-selection task using NASA-CRM. The goal is to select among 44 held-out configurations whose mean pressure distribution best matches a prescribed target under uncertain angle of attack. We treat AoA as uniform over $[-2.25^\circ, 7.25^\circ]$ and select the configuration with the lowest relative ℓ_2 error between its predicted mean pressure distribution and the target.

Evaluation We compare NSF with MC using $K \in \{2, 4, 8, 16, 32, 64, 128, 256, 512\}$. Reference mean pressure distributions are computed by dense inference with the pretrained Transolver. MC experiments are repeated over three random seeds. Table 5 reports the reference-based target mismatch of the selected configuration, how often each method selects the reference-best configuration, and the total inference time across all 44 candidates.

Table 5: Aerodynamic selection: NSF versus representative MC budgets. Mismatch is evaluated against the dense teacher reference for the selected configuration; MC values and times are averaged over three seeds. NSF selection is deterministic.

Method	Target mismatch ↓	Best selected	Aggregate time (s)
NSF (SG)	0.06884	Yes	3.32
MC-2	0.17131	1/3	3.32
MC-128	0.11108	1/3	160.12
MC-256	0.07844	2/3	320.17
MC-512	0.06884	3/3	640.25

Results NSF selects the reference-best configuration using two endpoint evaluations per candidate. MC requires $K = 512$ to select the same configuration in all three trials, the first tested budget to do so. At this budget, MC takes 640.25 seconds across all candidates, compared with 3.32 seconds for NSF, corresponding to approximately $193\times$ lower aggregate inference time.

B.4 Full statistics for benchmarks

As a supplement to Figure 2, we provide the remaining evaluation of all three statistics here. Specifically, the reduction in the number of model inferences is tested under the largest condition interval.

B.5 Model diagnostics and ablations

Per-Trajectory diagnostics for the 2D dynamical system To further inspect what neural statistical functions learn beyond interval-level error metrics, we visualize per-trajectory diagnostics on several held-out trajectories from the 2D dynamical-system experiment. For each selected trajectory, we compare both the learned prefix statistic and the corresponding reconstructed local signal against

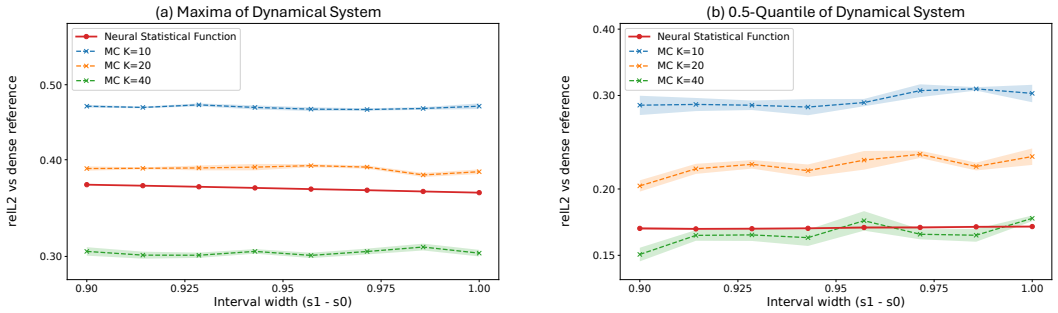

Figure 9: Maxima and 0.5-quantiles of the 2D dynamical system. Reference is dense MC of single-condition model.

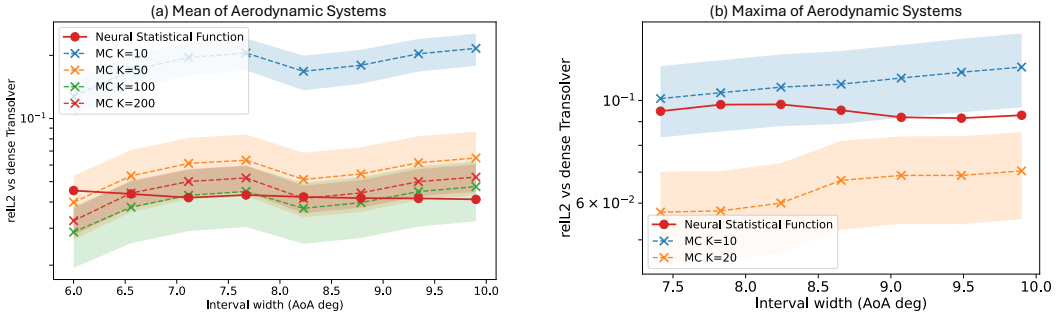

Figure 10: Mean and maxima of aerodynamics. Reference is dense MC of single-condition model.

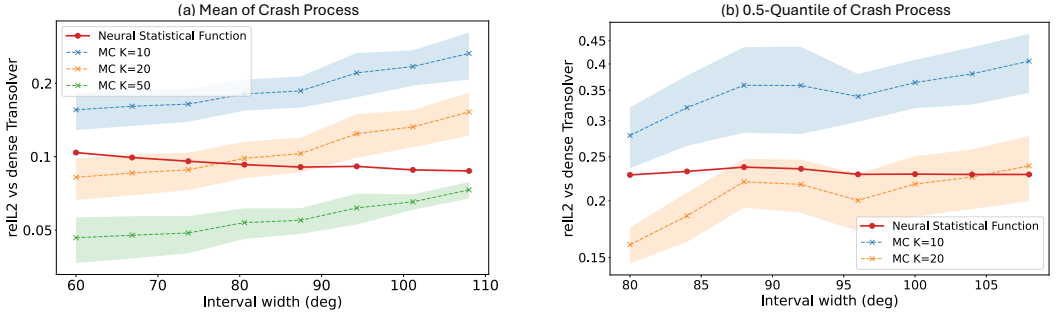

Figure 11: Mean and 0.5-quantile of crash. Reference is dense MC of single-condition model.

their reference values. Specifically, the left column of Figure 12 shows the learned prefix statistic $\tilde{\boldsymbol{h}}_\theta(\mathbf{x}, s)$ against the analytic reference

$$\tilde{\boldsymbol{h}}(\mathbf{x}, s) = \frac{1}{s} \int_0^s \boldsymbol{h}(\mathbf{x}, t)\, \mathrm{d}t,$$

while the right column compares the reconstructed local signal

$$\frac{\mathrm{d}}{\mathrm{d}s}\Big[s\, \tilde{\boldsymbol{h}}_\theta(\mathbf{x}, s)\Big]$$

with the exact instantaneous response $\boldsymbol{\psi}(\mathbf{x}, s) = \boldsymbol{h}(\mathbf{x}, s)$. The reconstruction is obtained by finite differences in s, matching the implementation used during training. These diagnostics complement the interval-query results in Figure 4. Agreement in the prefix statistic confirms that endpoint evaluations of $\tilde{\boldsymbol{h}}_\theta$ recover interval means accurately, while agreement in the reconstructed local signal verifies that the learned surrogate approximately satisfies the first-order identity in Eq. (13).

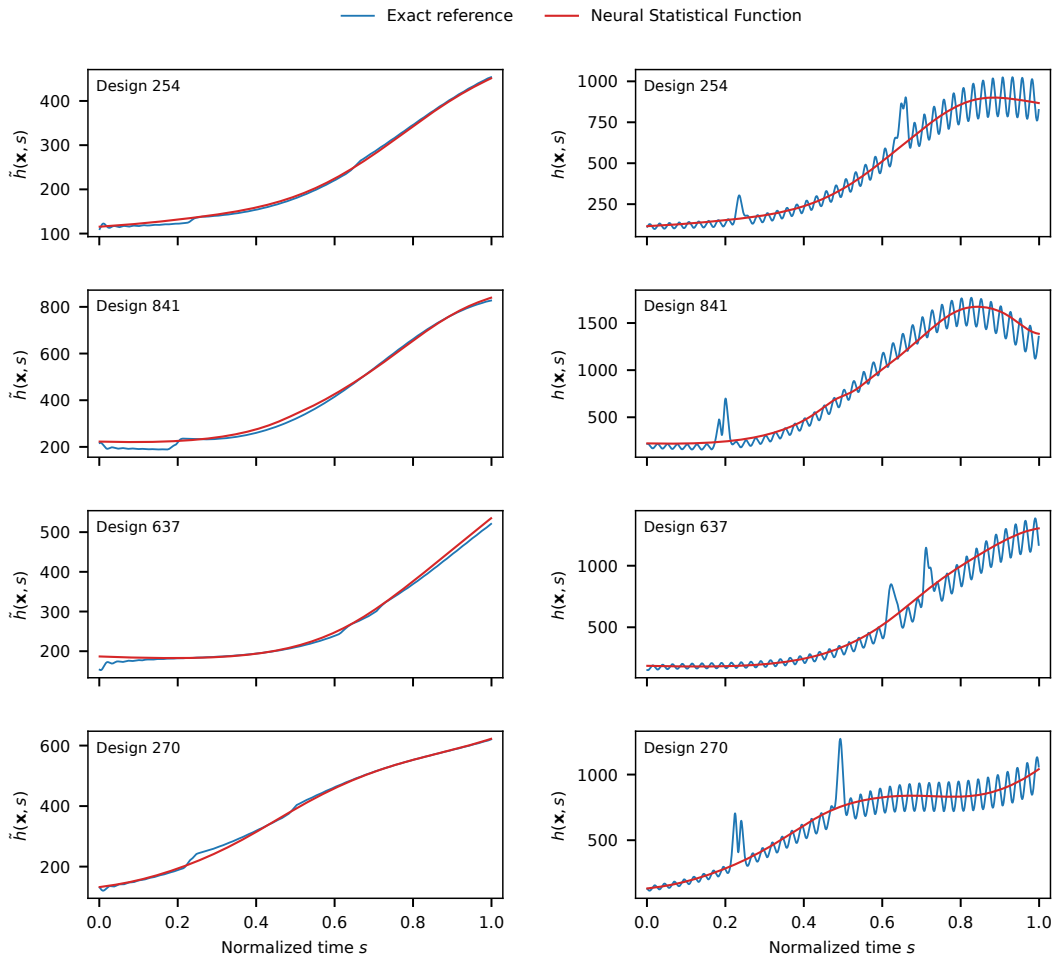

Figure 12: 2D dynamical-system diagnostics for selected illustrative held-out designs, using the same seed-0 NSF as Figure 4(a). Left: learned prefix mean versus its analytic reference. Right: reconstructed local signal versus the exact instantaneous response.

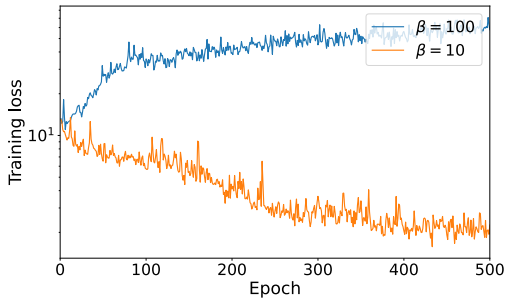

Figure 13: Training loss over the first 500 epochs for β=100 and β=10.

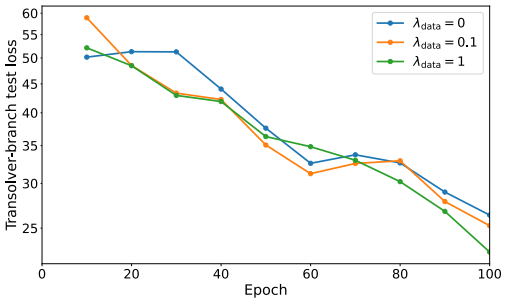

Figure 14: $\mathcal{L}_{\text{neural}}$ on test set over the first 100 epochs for $\lambda_{\text{data}} = 0$, $\lambda_{\text{data}} = 0.1$, $\lambda_{\text{data}} = 1$.

CDF inversion diagnostic The learned CDF is not constrained to be globally monotone. The running monotone envelope described in Appendix C stabilizes updates over the thresholds visited during bisection. As a diagnostic, projecting the clipped raw CDF onto a nondecreasing isotonic curve on a 33-point threshold grid gives a median per-node Pearson correlation of 0.968 between the two curves.

Ablations of β on Car-Crash maxima The log-sum-exp approximation in Eq. (7) introduces a smoothing temperature β. Increasing β reduces the bias between the soft maximum and the hard maximum, but it also amplifies variation in the transformed local signal $\boldsymbol{\psi}(\mathbf{x}, s) = \exp(\beta \boldsymbol{h}(\mathbf{x}, s))$. In particular, large β concentrates the interval statistic on a small subset of high-stress angles, so small errors in the single-sample response or in the finite-difference estimate of $\partial_s \tilde{\boldsymbol{h}}_\theta$ can produce large changes in the identity target. This leads to higher-variance gradients and less stable optimization. In our experiments, large β values produced noisier convergence and occasional loss spikes, whereas smaller β values were easier to train but yielded an overly smoothed maximum. Based on this trade-off, we set $\beta = 10$ for the experiments.

Ablations on λ_{data} in hybrid supervision We use $\lambda_{\text{data}} = 0.1$ as the default setting without extensive hyperparameter tuning and observe a clear improvement. To examine the effect of this weight, we conduct an ablation by reporting the Transolver-branch test loss $\mathcal{L}_{\text{neural}}$ when training neural statistical functions for maximum prediction in the crash process, as illustrated in Figure 14.

The loss curves show that including the data loss term ($\lambda_{\text{data}} > 0$) improves optimization of the neural branch, indicating that the neural and data losses provide complementary supervision. From the loss curve in Figure 14, we can observe that adjusting λ_{data}, *e.g.*, $\lambda_{\text{data}} = 1$, may further reduce the test loss and improve the model performance. Since we focus primarily on the methodology in this paper, we leave detailed hyperparameter tuning to future work, where some adaptive hyperparameter tuning techniques [16] can be useful.

Training objective ablation We compare stop-gradient training with direct minimization of the squared residual of Eq. (13), using the same prefix architecture. Direct residual improves NASA mean accuracy, whereas stop-gradient requires less training time per step and peak memory (Table 6).

Table 6: Training objective ablation on NASA interval-mean prediction.

Objective	Interval relative ℓ_2	Step time (ms)	Peak memory (MiB)
Stop-gradient	0.05738	**78.08**	**362.6**
Direct residual	**0.04083**	97.39	984.6

B.6 Robustness to imperfect teachers

We test moderate observation corruption in NASA mean prediction. In the noise setting, each training pressure label receives one fixed, independent, zero-mean Gaussian perturbation with standard deviation equal to 5% of the training-response standard deviation. In the missing-label setting, a fixed, configuration-specific mask removes 15% of spatial pressure labels throughout training. For each setting, we retrain the single-condition Transolver, freeze it, and then train NSF from that teacher. MC uses the same teacher. The clean NSF result uses direct-residual training.

Table 7: NASA mean prediction under noisy or incomplete teacher-training observations. Parentheses give logical model evaluations. Errors are measured against the clean teacher's interval-mean reference.

Training observations	NSF (2)	MC (5)	MC (10)	MC (20)	MC (50)	MC (100)
Clean	**0.04083**	0.23989	0.17145	0.11618	0.07515	0.05352
Fixed 5% Gaussian noise	**0.07238**	0.25558	0.19213	0.14341	0.11051	0.09503
Persistent 15% missing labels	**0.07198**	0.25630	0.19236	0.14290	0.10964	0.09432

Two-evaluation NSF remains stable and achieves approximately 24% lower error than MC-100 under both corruptions (Table 7). To separate approximation of a teacher from the teacher's own error, Table 8 also reports agreement with each model's own teacher. NSF approximates the corrupted teacher means with errors of 0.047–0.049, close to its clean result. Its error against the clean teacher increases as the corrupted teacher's means shift. Direct simulation supervision can mitigate this dependence, as illustrated by the crash hybrid-supervision results, but does not automatically eliminate systematic teacher bias.

Table 8: Teacher-error decomposition for NASA interval means. "Clean" denotes the clean teacher. NSF shift compares corrupted-teacher NSF predictions with clean-teacher NSF predictions.

Training pipeline	Teacher curve vs. clean	Teacher mean vs. clean	NSF vs. own teacher	NSF vs. clean	NSF shift
Clean	0	0	0.04083	0.04083	0
Fixed 5% label noise	0.12975	0.07327	0.04898	0.07238	0.06700
Persistent 15% missing labels	0.12592	0.07216	0.04739	0.07198	0.06791

B.7 Computational cost and deployment

Offline training and amortization NSF trades a one-time training cost for lower repeated-query latency. Training for $2{,}000$ epochs takes 5.46 hours on four GPUs, approximately 21.8 GPU-hours. The pretrained single-condition Transolver is shared by NSF and MC, so its training is a common cost. Table 9 summarizes NASA interval-mean query latency. Using these MC costs, the accumulated query-time savings offset the 5.46-hour training elapsed time after approximately $15{,}200$ or $7{,}500$ mean queries relative to MC-100 or MC-200, respectively.

Table 9: NASA interval-mean query latency.

Method	Latency (ms)	Relative latency
NSF	**22.14**	$1.00\times$
MC-100	1317.99	$59.53\times$
MC-200	2634.88	$119.01\times$

Dense caching Precomputation is a relevant alternative for a small, fixed catalog of geometries. A cache of M FP32 fields at N nodes for each of G geometries occupies $4NGM$ bytes. At the $N = 32{,}000$ resolution used in Table 3, caching 1,000 fields requires approximately 0.119 GiB per geometry, 5.25 GiB for 44 geometries, and 119.2 GiB for 1,000 geometries. Larger spatial meshes increase these costs proportionally. The NSF checkpoint is 18.3 MB; its persistent model storage does not grow with the number of future query geometries. A cache must be populated for each new geometry, whereas NSF directly evaluates its learned continuous prefix representation. Caching can be preferable for a small fixed catalog; NSF is attractive for new geometries, fine condition resolution, and storage-constrained repeated inference.

B.8 Quantitative results

In Tables 10-14, we present numerical values of plotted main text results for quantitative comparison.

Table 10: Mean relative ℓ_2 error for the 2D dynamical system versus the independent 1,000-sample MC reference, corresponding to Figure 4(b). Parentheses give evaluation counts.

Width $s_1 - s_0$	NSF (2)	IID MC (50)	IID MC (100)	IID MC (200)	IID MC (400)	Midpoint (40)	Gauss–Legendre (40)	Scrambled Sobol (40)	Stratified (40)
0.9000	0.0298	0.0881 ±0.0014	0.0625 ±0.0013	0.0449 ±0.0003	0.0356 ±0.0004	0.0263	0.0383	0.0376 ±0.0032	0.0355 ±0.0016
0.9143	0.0279	0.0878 ±0.0012	0.0625 ±0.0009	0.0470 ±0.0005	0.0349 ±0.0002	0.0277	0.0317	0.0372 ±0.0028	0.0376 ±0.0025
0.9286	0.0281	0.0885 ±0.0010	0.0620 ±0.0010	0.0463 ±0.0005	0.0350 ±0.0006	0.0283	0.0357	0.0359 ±0.0018	0.0345 ±0.0014
0.9429	0.0282	0.0855 ±0.0009	0.0625 ±0.0005	0.0460 ±0.0009	0.0350 ±0.0005	0.0306	0.0367	0.0352 ±0.0016	0.0326 ±0.0008
0.9571	0.0289	0.0868 ±0.0014	0.0637 ±0.0009	0.0469 ±0.0009	0.0362 ±0.0009	0.0362	0.0317	0.0385 ±0.0027	0.0335 ±0.0009
0.9714	0.0292	0.0900 ±0.0010	0.0657 ±0.0018	0.0480 ±0.0008	0.0356 ±0.0004	0.0450	0.0395	0.0401 ±0.0031	0.0332 ±0.0011
0.9857	0.0283	0.0879 ±0.0010	0.0649 ±0.0010	0.0475 ±0.0010	0.0372 ±0.0007	0.0911	0.0333	0.0421 ±0.0050	0.0340 ±0.0011
1.0000	0.0281	0.0857 ±0.0018	0.0640 ±0.0004	0.0476 ±0.0013	0.0347 ±0.0005	0.1421	0.0385	0.0392 ±0.0034	0.0345 ±0.0010

Table 11: Mean relative ℓ_2 error for the 2D dynamical system versus the analytic interval mean, corresponding to Figure 4(c). Parentheses give evaluation counts.

Width $s_1 - s_0$	NSF (2)	IID MC (50)	IID MC (100)	IID MC (200)	IID MC (400)	Midpoint (40)	Gauss–Legendre (40)	Scrambled Sobol (40)	Stratified (40)
0.9000	0.0220	0.0859 ±0.0009	0.0600 ±0.0010	0.0415 ±0.0006	0.0302 ±0.0005	0.0194	0.0338	0.0323 ±0.0036	0.0296 ±0.0016
0.9143	0.0217	0.0858 ±0.0013	0.0600 ±0.0006	0.0430 ±0.0007	0.0301 ±0.0003	0.0215	0.0263	0.0322 ±0.0032	0.0327 ±0.0027
0.9286	0.0214	0.0865 ±0.0010	0.0593 ±0.0012	0.0423 ±0.0007	0.0301 ±0.0004	0.0213	0.0308	0.0307 ±0.0019	0.0291 ±0.0016
0.9429	0.0212	0.0837 ±0.0007	0.0598 ±0.0008	0.0420 ±0.0006	0.0302 ±0.0005	0.0241	0.0312	0.0299 ±0.0020	0.0273 ±0.0007
0.9571	0.0210	0.0851 ±0.0013	0.0611 ±0.0010	0.0427 ±0.0010	0.0302 ±0.0007	0.0307	0.0258	0.0337 ±0.0031	0.0279 ±0.0009
0.9714	0.0209	0.0881 ±0.0009	0.0625 ±0.0019	0.0429 ±0.0009	0.0295 ±0.0005	0.0395	0.0329	0.0341 ±0.0038	0.0265 ±0.0015
0.9857	0.0208	0.0857 ±0.0010	0.0620 ±0.0008	0.0432 ±0.0008	0.0312 ±0.0006	0.0879	0.0258	0.0365 ±0.0057	0.0269 ±0.0012
1.0000	0.0206	0.0829 ±0.0017	0.0612 ±0.0008	0.0437 ±0.0008	0.0302 ±0.0009	0.1398	0.0324	0.0343 ±0.0037	0.0289 ±0.0012

Table 12: NASA-CRM quantile field $\mathrm{relL2}$ versus dense Transolver reference. Neural statistical function vs. MC at $K{=}10$ and $K{=}20$ Transolver calls. MC entries are mean $\pm$ 1 SE over 5 seeds.

α-Quantile	Interval width (deg)	Neural statistical function	MC $K{=}10$	MC $K{=}20$
0.5	8.000	0.199	0.441 ± 0.083	0.242 ± 0.036
	8.475	0.201	0.533 ± 0.091	0.268 ± 0.037
	8.950	0.205	0.574 ± 0.096	0.282 ± 0.038
	9.425	0.204	0.576 ± 0.097	0.292 ± 0.040
	9.900	0.203	0.563 ± 0.097	0.297 ± 0.042
0.7	8.000	0.235	0.328 ± 0.081	0.154 ± 0.039
	8.475	0.242	0.346 ± 0.084	0.164 ± 0.037
	8.950	0.244	0.361 ± 0.086	0.172 ± 0.038
	9.425	0.248	0.380 ± 0.093	0.178 ± 0.039
	9.900	0.249	0.399 ± 0.099	0.185 ± 0.041
0.9	8.000	0.211	0.213 ± 0.064	0.096 ± 0.014
	8.475	0.180	0.199 ± 0.058	0.090 ± 0.013
	8.950	0.155	0.201 ± 0.057	0.091 ± 0.012
	9.425	0.145	0.214 ± 0.060	0.096 ± 0.013
	9.900	0.144	0.231 ± 0.066	0.102 ± 0.014

Table 13: Crash max $\mathrm{relL2}$ versus interval width, evaluated against dense Transolver reference. Neural statistical function with and without data cosupervision are compared with MC at $K{=}5$ and $K{=}10$ solver calls. MC entries are mean $\pm$ 1 SE.

Width (deg)	NSF with data supervision	NSF without data supervision	MC $K{=}5$	MC $K{=}10$
80.0	0.323	0.352	0.319 ± 0.011	0.275 ± 0.014
84.0	0.326	0.337	0.325 ± 0.017	0.280 ± 0.020
88.0	0.329	0.340	0.335 ± 0.021	0.288 ± 0.026
92.0	0.330	0.341	0.342 ± 0.023	0.295 ± 0.028
96.0	0.332	0.343	0.362 ± 0.022	0.312 ± 0.021
100.0	0.335	0.346	0.373 ± 0.025	0.322 ± 0.023
104.0	0.335	0.346	0.387 ± 0.030	0.335 ± 0.028
108.0	0.338	0.349	0.398 ± 0.033	0.347 ± 0.032

Table 14: Crash max $\mathrm{relL2}$ versus interval width, evaluated against ground truth data. Neural statistical functions with and without data cosupervision are compared with MC at $K{=}5$ and $K{=}10$ solver calls. MC entries are mean $\pm$ 1 SE.

Width (deg)	NSF with data supervision	NSF without data supervision	MC $K{=}5$	MC $K{=}10$
80.0	0.452	0.458	0.456 ± 0.009	0.426 ± 0.008
84.0	0.449	0.459	0.459 ± 0.014	0.428 ± 0.016
88.0	0.451	0.461	0.464 ± 0.016	0.434 ± 0.020
92.0	0.453	0.462	0.472 ± 0.019	0.440 ± 0.022
96.0	0.454	0.463	0.488 ± 0.017	0.451 ± 0.014
100.0	0.456	0.465	0.496 ± 0.019	0.458 ± 0.016
104.0	0.457	0.466	0.506 ± 0.023	0.468 ± 0.021
108.0	0.459	0.468	0.514 ± 0.026	0.476 ± 0.024

C Implementation details

2D dynamical system The trajectories in Section 4.1 are $\mathbf{y}(s) = r(s)(\cos\eta(s), \sin\eta(s))$, where

$$r(s) = a + bs + 0.1\sin(2\pi s + \phi_r) + 0.25\gamma_r \exp\left[-\tfrac{1}{2}\left(\frac{s-\mu_r}{0.008}\right)^2\right], \tag{23}$$

$$\eta'(s) = \omega_0 + 2\pi\alpha\cos(2\pi s + \phi_\omega) + 1.5\gamma_\omega \sin(80\pi s + \phi_h) + 8.0\gamma_\omega \exp\left[-\tfrac{1}{2}\left(\frac{s-\mu_\omega}{0.008}\right)^2\right]. \tag{24}$$

We draw $a \sim \mathrm{Unif}(0.25, 1)$, $b \sim \mathrm{Unif}(0.25, 1.5)$, $\omega_0 \sim \mathrm{Unif}(2\pi, 6\pi)$, and $\alpha \mid \omega_0 \sim \mathrm{Unif}(0, 0.35\omega_0/(2\pi))$, with independent phases uniform on $[0, 2\pi]$, transient centers $\mu_r, \mu_\omega \sim \mathrm{Unif}(0.08, 0.92)$, and amplitudes $\gamma_r, \gamma_\omega \sim \mathrm{Unif}(0.65, 1.35)$.

Evaluation protocol We evaluate all methods on held-out test cases using width-binned relative ℓ_2 error. All MC results are averaged over five random seeds and reported as mean $\pm$ one standard error.

Dynamical-system evaluation For the 2D dynamical-system mean experiment (Section 4.1), we evaluate eight equally spaced widths in $[0.90, 1.00]$ on 900 held-out designs, using two interval positions per width except for one full-domain query, for 13,500 queries in total.

Aerodynamic evaluation For NASA-CRM (Section 4.2), we evaluate quantile levels $\alpha \in \{0.5, 0.7, 0.9\}$ over AoA interval widths from 8.0° to 9.9°, with a dense-grid Transolver reference computed from uniformly spaced AoA evaluations and MC baselines at $K \in \{10, 20\}$. Neural statistical function quantiles are recovered by five bisection steps per node with smoothing temperature $\varepsilon = 10^{-2}$; to improve robustness to local non-monotonicities in the learned CDF, bisection uses a running monotone envelope of the predicted CDF values for bracket updates.

Crash evaluation For the crash experiment (Section 4.3), we evaluate maximum-stress prediction over impact-angle widths from 80° to 108°, using both dense Transolver and data-derived simulation references, and compare with MC baselines at $K \in \{5, 10\}$.

D Broader impacts

This paper presents neural statistical functions as a new family of neural networks, whose design principle can be inspiring for future research. Besides, the proposed model can significantly accelerate statistical estimation over a range of varying conditions. One potential application is industrial design, where conventional neural surrogates typically provide predictions under a specific condition, while neural statistical functions allow engineers to directly estimate distributional statistics under diverse operating scenarios. This capability may reduce design-loop latency, improve uncertainty-aware evaluation, and lower the computational cost of large-scale simulation-based analysis. Beyond industrial design, the proposed framework may also benefit decision-making scenarios that require fast statistical estimates, such as motion planning with total energy estimation.

This paper mainly focuses on the scientific problem and we are fully committed to ensuring ethical considerations are taken into account. Thus, we believe there are no potential ethical risks.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: This paper presents neural statistical functions as a new framework for fast statistics estimation, supported by Remark 1 and the experiments summarized in Figure 2.

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: We discuss limitations in Section 5.

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [Yes]

Justification: The proofs are provided in Appendix A.

Guidelines:

- The answer [N/A] means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: We include setup descriptions in the main text and Appendix C.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
     (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [No]

Justification: Data for dynamical and aerodynamic systems is public. We plan to release the code and car-crash simulation data.

Guidelines:

- The answer [N/A] means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

   Answer: [Yes]

   Justification: Training and test settings appear in Section 4 and Appendix C.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

   Justification: MC experiments are repeated over five random seeds. Error bands indicate one standard error, as described in Section 4 and Appendix C.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
   - The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
   - The assumptions made should be given (e.g., Normally distributed errors).

- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
- If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: We have included in "implementation details" of main text and appendix.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: We confirm that the research conforms to the NeurIPS Code of Ethics.

   Guidelines:

   - The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: We discuss broader impacts in Appendix D.

    Guidelines:

    - The answer [N/A] means that there is no societal impact of the work performed.
    - If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.

- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

    Answer: [N/A]

    Justification: Not applicable.

    Guidelines:

    - The answer [N/A] means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: All assets used are public with a proper license.

    Guidelines:

    - The answer [N/A] means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
    - For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.

- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [Yes]

    Justification: We provide detailed benchmark descriptions.

    Guidelines:

    - The answer [N/A] means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [N/A]

    Justification: The research does not involve crowdsourcing or human subjects.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [N/A]

    Justification: The research does not involve human subjects.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

Answer: [N/A]

Justification: LLMs are not used in the core methods of this paper.

Guidelines:

- The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
- Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.