DEBIASED NEURAL OPERATORS

Anonymous authors
Paper under double-blind review

ABSTRACT

Neural operators are widely used to approximate solution maps of complex physical systems. In many applications, however, the goal is not to recover the full solution trajectory, but to summarize the solution trajectory via a scalar target quantity (e.g., a functional such as time spent in a target range, time above a threshold, accumulated cost, or total energy). In this paper, we introduce DOPE (*debiased neural operator*): a semiparametric estimator for such target quantities of solution trajectories obtained from neural operators. DOPE is broadly applicable to settings with both partial and irregular observations and can be combined with arbitrary neural operator architectures. We make three main contributions. (1) We show that, in contrast to DOPE, naïve plug-in estimation can suffer from first-order bias. (2) To address this, we derive a novel one-step, Neyman-orthogonal estimator that treats the neural operator as a high-dimensional nuisance mapping between function spaces, and removes the leading bias term. For this, DOPE uses a weighting mechanism that simultaneously accounts for irregular observation designs and for how sensitive the target quantity is to perturbations of the underlying trajectory. (3) To learn the weights, we extend automatic debiased machine learning to *operator-valued nuisances* via Riesz regression. We demonstrate the benefits of DOPE across various numerical experiments.

1 INTRODUCTION

Neural operators (Kovachki et al., 2023; Li et al., 2021; Lu et al., 2021) are a powerful framework for learning solution maps of complex physical systems governed by partial differential equations (PDEs). In particular, neural operators can efficiently learn mesh-independent approximations of PDE solutions that generalize across varying input functions, including different boundary conditions and forcing terms (Azizzadenesheli et al., 2024; Rosofsky et al., 2023). As a result, they are increasingly used in various applications such as scientific computing (Azizzadenesheli et al., 2024; Cao et al., 2024; Faroughi et al., 2024), climate science (Kurth et al., 2023; Peng et al., 2024; Wen et al., 2023), and engineering (Kobayashi et al., 2024; Shukla et al., 2024; Wang et al., 2025).

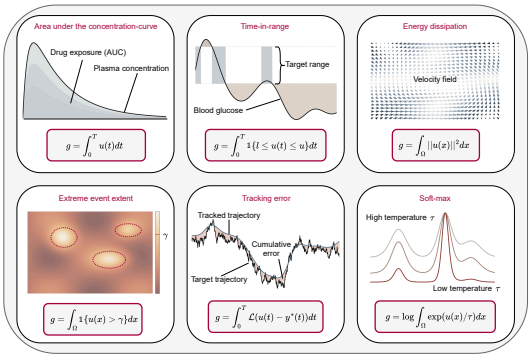

Figure 1: **Examples:** In many scientific applications, solution trajectories are summarized by scalar quantities (i.e., functionals g).

In this paper, ***we aim to learn a scalar target quantity (i.e., a functional) of the solution trajectory***. This is relevant in many scientific applications, where the main quantity of interest is not the full solution trajectory produced by a PDE solver or its neural operator approximation. Instead, the aim is often to summarize the trajectory into a scalar target quantity that is directly relevant for decision-making or scientific interpretation (see Figure 1).

Examples: (1) In medicine, physicians care about whether a patient's vital signs remain within a target range, such as maintaining blood glucose above a critical threshold or keeping heart rate and respiratory rate within a safe range (Hashiguchi et al., 2023; Iqbal et al., 2022; Yang & Yang, 2024; Zhou et al., 2025). (2) In climate modeling, researchers quantify whether forecasts exceed critical thresholds, such as extreme temperatures (Price et al., 2025) (3) In fluid dynamics and energy systems,

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

the goal is to measure total fluxes or energy over time (Grachev et al., 2020). ④ In control systems, engineers quantify the accumulated tracking error or other stability metrics over time (Dong et al., 2025; Widdershins et al., 2026). Importantly, such functionals typically depend on the trajectory in complex, nonlinear ways.

Why a naïve estimation is problematic: A naïve strategy for estimating such functionals is the following (which we later refer to as *plug-in estimator*): first train a neural operator, then evaluate the functional of interest on the predicted trajectory, and use the result as the final estimate. However, we later show that this naïve plug-in estimator can be systematically biased, even when the neural operator predicts the solution trajectory reasonably well. The reason is that the quantity of interest is a nonlinear summary of the trajectory, so approximation errors in the learned solution map do not simply average out but instead propagate through the functional and thus accumulate into a non-negligible error in the final estimate. We refer to this systematic error as *plug-in bias*.[1].

In this paper, we introduce the *debiased neural operator* (DOPE): a semiparametric framework for estimating scalar summaries of solution trajectories with neural operators. Our DOPE framework is broadly applicable to settings with both partial and irregular observations. Further, DOPE is model-agnostic and can be instantiated with *any* neural operator architecture (e.g., such as Fourier neural operators (Li et al., 2021) or DeepONet (Lu et al., 2021)), and is therefore broadly applicable across scientific applications.

To construct DOPE, we derive a novel one-step, Neyman-orthogonal estimator that treats the neural operator as a high-dimensional nuisance mapping between function spaces. Specifically, we use the Riesz representer to build a score that is first-order insensitive to errors in the learned neural operator. As a result, approximation errors of the neural operator affect the estimated functional only through higher-order terms, and thus remove the plug-in bias. Methodologically, DOPE corrects the neural-operator-induced bias using pointwise residuals of the solution trajectory, and, for this, uses a weighting mechanism that simultaneously accounts for irregular observation designs and for how sensitive the target quantity is to perturbations of the underlying trajectory. To learn the weights, we propose a tailored automatic debiasing approach based on Riesz regression and automatic differentiation.

Contributions. In summary, our contributions are as follows:[2]

① We develop a novel semiparametric framework called DOPE for inference on scalar functionals of neural operator outputs. Crucially, DOPE removes the plug-in bias of naïve estimation strategies.

② We further develop an automatic debiasing procedure for operator-valued nuisance functions via Riesz regression and automatic differentiation. In this way, we provide a non-trivial extension of automatic debiased machine learning (DML) (Chernozhukov et al., 2021; 2022a) from *function-valued* to *operator-valued* nuisances.

③ We establish various theoretical properties of DOPE, including asymptotic normality and confidence intervals (under standard regularity conditions). Further, our framework easily integrates prediction-powered inference (PPI) (Angelopoulos et al., 2023a;b) for neural operators.

2 Related Work

• **Plug-in bias:** In many statistical problems, estimating the target parameter of interest requires first estimating nuisance functions (i.e., auxiliary functions that are not themselves of direct interest). Orthogonal estimation (Foster & Syrgkanis, 2023; Morzywolek et al., 2023) constructs estimators so that small errors in these nuisance functions affect the target only at second order. A common issue is that errors in these nuisance estimates propagate into the final estimate, often leading to systematic bias—commonly referred to as *plug-in bias* (Curth et al., 2020; Kennedy, 2022).

Orthogonal estimation (Foster & Syrgkanis, 2023; Morzywolek et al., 2023) addresses the problem of plug-in bias by constructing estimators that are insensitive to small errors in the nuisance functions.

[1]Plug-in bias is a know problem in other fields such as causal machine learning (Curth et al., 2020; Kennedy, 2022). Here, we extend the idea to neural operators, yet which comes with new challenges as we move from *function-valued* nuisances to *operator-valued* nuisances.

[2]Code is available at `https://anonymous.4open.science/r/1C71`.

Intuitively, this means that approximation errors do not propagate as first-order biases, but only at second order in the target quantity. In semiparametric theory, this is typically achieved by deriving influence-function-based moment conditions (Hines et al., 2022; Ichimura & Newey, 2022). To remove the first-order bias, one then aims to derive a one-step estimator (Kennedy, 2022), which "corrects" (i.e., debiases) the plug-in estimate with the sample average of a debiasing score, and thereby removes the first-order bias term.

However, the problem of plug-in bias has been studied in several settings, such as for causal machine learning (Curth et al., 2020; Kennedy, 2022), but **not** for neural operators. Importantly, existing works typically focus on settings that have *function-valued* nuisances (e.g., Hess et al., 2026a;b; Melnychuk et al., 2024). ⇒ *In contrast, our setting involves* ***operator-valued*** *nuisances, which introduces new challenges and requires tailored estimation techniques.*

• **Automatic debiasing:** Orthogonal estimation has attractive second-order bias properties, but applying it often requires deriving the *debiasing score* analytically. A remedy is automatic debiasing, which aims to construct this correction directly from the target functional (Carone et al., 2019; Luedtke, 2024).

This idea can be implemented via automatic debiased machine learning, which reduces debiasing to estimation of an auxiliary function through Riesz regression rather than requiring a closed-form derivation of the correction term (Chernozhukov et al., 2021; 2022b). Prior work studies this idea for *function-valued* nuisances using random forests and multi-layer perceptrons (Chernozhukov et al., 2022a), for regularized Riesz regression (Chernozhukov et al., 2022c), under covariate shift (Chernozhukov et al., 2023), and for smooth functionals of M-estimands (van der Laan et al., 2025). ⇒ *In contrast, we extend automatic debiasing to* ***operator-valued*** *nuisance functions.*

3 Problem Formulation

• **PDE setup:** We consider the standard setting of operator learning (Kovachki et al., 2023). Let $\Omega \subset \mathbb{R}^d$ be a bounded domain.[3] Let $\mathcal{A}$ denote a Banach space of admissible input functions (e.g., coefficient functions, boundary conditions, or forcing terms). For each admissible coefficient function $a \in \mathcal{A}$, we consider a PDE of the generic form

$$\mathcal{T}_a u = f, \tag{1}$$

with homogeneous boundary conditions. We assume that, for each $a \in \mathcal{A}$, the PDE admits a unique weak solution

$$\mathcal{T}_a^{-1} f = u \in \mathcal{U}, \tag{2}$$

where, throughout, we take $\mathcal{U} = L^2(\Omega, \mu)$, which is the Hilbert space of square-integrable functions on Ω with inner product induced by Lebesgue measure μ. We denote by

$$S_0 : \mathcal{A} \to \mathcal{U}, \qquad S_0(a) = u(a, \cdot), \tag{3}$$

the *unknown solution operator*, which maps input functions to solution trajectories.

• **Neural operators:** Neural operators are learned approximations of the solution operator S_0 (Kovachki et al., 2021; Li et al., 2021; Lu et al., 2021). Given training data consisting of input functions and partial solution observations, neural operators learn a mapping $\hat{S} : \mathcal{A} \to \mathcal{U}$, such that $\hat{S}(a) \approx S_0(a)$. They operate directly on functional inputs and outputs defined on the continuous domain Ω, which can be evaluated at arbitrary locations $x \in \Omega$. We provide an extended related work on neural operators in Supplement A.

• **Partial and irregular measurements:** In many scientific workflows, *a central challenge* is that the evaluation locations depend on the input through measurement protocols, adaptive solvers, or experimental constraints (Berg & Nyström, 2019; Zhou et al., 2025). Hence, solution trajectories are often observed only *partially and irregularly* (Chen et al., 2018; Hess & Feuerriegel, 2025; Hess et al., 2024; Iakovlev et al., 2023). Accordingly, let A denote a random coefficient function with associated solution trajectory $S_0(A) \in \mathcal{U}$. Rather than observing the full trajectory, we observe noisy point evaluations

$$Y_k = S_0(A)(X_k) + \varepsilon_k, \qquad k = 1, \ldots, K, \tag{4}$$

[3] We make no formal distinction between time and spatial domain, and w.l.o.g. Ω can refer to both.

where $K \in \mathbb{N}$ may be random and depend on A, where $X_k \in \Omega$ is an evaluation location, and where the noise satisfies $\mathbb{E}[\varepsilon_k \mid A, X_k] = 0$ and $\mathbb{E}[\varepsilon_k^2 \mid A, X_k] < \infty$. We model the sampling mechanism through a joint law $\mathbb{P}$ on $\mathcal{A} \times \Omega$ with factorization $\mathbb{P}(\mathrm{d}a, \mathrm{d}x) = \mathbb{P}(\mathrm{d}x \mid a)\,\mathbb{P}(\mathrm{d}a)$, where $\mathbb{P}(\cdot \mid a)$ is an input-dependent design distribution on Ω. Given (A, K), we assume that the pairs $\{(X_k, Y_k)\}_{k=1}^{K}$ are conditionally i.i.d. draws from a generic pair (X, Y) with $X \mid A = a \sim \mathbb{P}(\cdot \mid a)$. We observe n i.i.d. realizations

$$O_i = \Big(A_i, K_i, \{(X_{ik}, Y_{ik})\}_{k=1}^{K_i}\Big), \qquad i = 1, \dots, n, \tag{5}$$

and write $\mathcal{D} = \{O_i\}_{i=1}^{n}$. Throughout, all $L^2(\mathbb{P})$ norms are taken with respect to the joint law of (A, X). This setting captures the common regime where generating new inputs is relatively cheap, while dense evaluations of the corresponding solution trajectory are costly (Meyer et al., 2023; Kim et al., 2025).

• **Identifiability:** In order to ensure identifiability from observations sampled under $\mathrm{d}\mathbb{P}$, we impose a standard design overlap assumption (Curth & van der Schaar, 2021; Hess & Feuerriegel, 2025).

Assumption 3.1 (Design overlap)**.** *Let μ denote a fixed reference measure on Ω (e.g., Lebesgue measure). Whenever it exists, we define the corresponding inverse design weight as*

$$\xi_0(a)(x) := \frac{\mathrm{d}\mu}{\mathrm{d}\mathbb{P}(\cdot \mid a)}(x). \tag{6}$$

Then, for $\mathbb{P}$-almost every $a \in \mathcal{A}$, the measure μ is absolutely continuous with respect to $\mathbb{P}(\cdot \mid a)$. Moreover, there exists a constant $C < \infty$ such that $0 < \xi_0(a)(x) \le C$ for $\mathbb{P}(\cdot \mid a)$-a.e. $x \in \Omega$.

Finally, for $\mathbb{P}$-almost every $a \in \mathcal{A}$, we define the conditional Hilbert space $\mathcal{H}_a := L^2(\Omega, \mathbb{P}(\cdot \mid a))$ with inner product $\langle f, h\rangle_a := \int_\Omega f(x)h(x)\ \mathrm{d}\mathbb{P}(x \mid a)$. Under Assumption 3.1, $\mathcal{H}_a$ embeds continuously into $\mathcal{U} = L^2(\Omega, \mu)$.

• **Objective:** Let $g : \mathcal{U} \to \mathbb{R}$ be our scalar-valued functional of interest. Throughout, we let g be twice Fréchet differentiable in a neighborhood of $\{S_0(A)\}$ with uniformly bounded second derivative, i.e.,

$$g(u + h) = g(u) + \mathrm{D}g_u(h) + R(u, h) \tag{7}$$

with $|R(u, h)| \lesssim \|h\|_{\mathcal{U}}^2$ for all u in this neighborhood and all sufficiently small h.[4] This is naturally satisfied by many common functionals in practice, including linear functionals, polynomial energy-type functionals, norm-based functionals, or weighted transport functionals.

Objective

We aim to perform inference on the averaged scalar functional

$$\theta := \mathbb{E}[g(S_0(A))]\,, \tag{8}$$

where the expectation is taken with respect to the population distribution of A.

Examples (see Figure 1)

• *Medicine.* A encodes subject- or drug-specific parameters (e.g., from pharmacokinetics); $S_0(A)(t)$ is the concentration–time curve; and $g(S_0(A))$ represents the time spent in a target range or above a clinically relevant threshold.
• *Fluid dynamics and energy systems.* A represents coefficients, forcing terms, or boundary conditions; $S_0(A)(x)$ is the resulting field; and $g(S_0(A))$ represents the flux over a spatial region or energy over time.
• *Control systems.* A parameterizes system dynamics or disturbances; $S_0(A)(t)$ is the resulting trajectory; and $g(S_0(A))$ represents accumulated tracking error or another scalar measure of stability and performance over time.

[4] We use standard notation $a(x) \lesssim b(x)$ to denote that there exists $C > 0$ such that $a(x) \le Cb(x)$ for all x.

By the Riesz representation theorem (Pavone, 1994), for every Fréchet differentiable functional $g : \mathcal{U} \to \mathbb{R}$ and every $u \in \mathcal{U}$, there exists a unique $w_g(u) \in \mathcal{U}$ such that

$$\mathrm{D}g_u(h) = \langle w_g(u), h \rangle_{\mathcal{U}} = \int_\Omega w_g(u)(x)\, h(x)\ \mathrm{d}\mu(x). \tag{9}$$

4 DOPE: Debiased estimation of functionals for neural operators

In the following, we focus on debiased estimation of the scalar functional in Equation 8. For this, we proceed as follows: (1) We first show why the naïve estimator leads to plug-in bias (§4.1). (2) We then introduce DOPE: a *one-step Neyman-orthogonal estimator* (see Figure 2). For this, we derive the *debiasing score* by expressing the Fréchet derivative as an inner product with a Riesz representer, which yields a first-order debiasing procedure (§4.2). (3) We show several desirable *theoretical properties* of DOPE (§4.3). (4) Finally, we present a tailored approach for *automatic DML with operator-valued nuisances* using the Riesz representer (§4.4). We provide the pseudo-code in Algorithm 1 in Supplement E.

4.1 Why naïve plug-in estimation is problematic

A naïve estimation strategy for Equation 8 is to estimate $S_0(A)$ with a neural operator $\hat{S}(A)$ and treat the surrogate as the ground-truth operator. However, such plug-in estimation leads to severe plug-in bias:

Proposition 4.1 (First-order bias of plug-in estimators)**.** *Let* $\Delta(\cdot) = \hat{S}(\cdot) - S_0(\cdot)$. *Then*

$$\left|\mathbb{E}[g(\hat{S}(A))] - \theta\right| \lesssim \left|\mathbb{E}\left[\mathrm{D}g_{S_0(A)}(\Delta(A))\right]\right| + \|\Delta\|^2_{L^2(\mathbb{P})}, \tag{10}$$

i.e., the neural operator error propagates linearly through the Fréchet differential, and induces ***first-order bias****.*

Proof. See Supplement B.1. □

Proposition 4.1 shows that the estimation error of the plug-in estimator is, in general, linear in $\mathbb{E}[\mathrm{D}g_{S_0(A)}(\Delta(A))]$ and thus does not vanish. As a result, root-n inference from the plug-in estimator requires this first-order bias to be negligible, i.e., of order $o(n^{-1/2})$. To understand what this implies, suppose the Fréchet derivative admits a uniformly bounded Riesz representer, so that $|\mathrm{D}g_{S_0(A)}(\Delta(A))| \lesssim \|\Delta(A)\|_{L^2(\mu)}$. Then, by overlap, this implies that the linear term can be controlled by $\left|\mathbb{E}[\mathrm{D}g_{S_0(A)}(\Delta(A))]\right| \lesssim \|\Delta\|_{L^2(\mathbb{P})}$. Hence, a sufficient condition for root-n inference is that the neural operator error satisfies $\|\Delta\|_{L^2(\mathbb{P})} = o(n^{-1/2})$. However, such rates are typically *highly unrealistic* in practice. Intuitively, the problem of the plug-in estimator is that nonlinear functionals propagate operator errors *linearly* through the Fréchet derivative, so even small approximation errors accumulate.

4.2 Debiased estimation via DOPE

As a remedy, we now derive a **one-step Neyman-orthogonal estimator** (DOPE). In contrast to the naïve plug-in estimator from Proposition 4.1, DOPE achieves root-n convergence under substantially weaker accuracy requirements on the neural operator.

The starting point is the leading bias term of the naïve plug-in estimator, $\mathbb{E}\left[\mathrm{D}g_{S_0(A)}\left(\Delta(A)\right)\right]$, where $\Delta(\cdot) = S_0(\cdot) - \hat{S}(\cdot)$ denotes the operator error. To eliminate the plug-in bias, we seek a correction term that *recovers the value of the functional derivative* $\mathrm{D}g_{S_0(A)}(\Delta(A))$ using observable quantities.

However, this is **non-trivial**: the perturbation $\Delta(A)$ is *not* observed as a function, but only through noisy point evaluations $Y_k - \hat{S}(A)(X_k)$ at locations X_k drawn from the sampling design. Hence, we can *not* directly evaluate the functional derivative.

Instead, we leverage that $\mathrm{D}g_{S_0(A)}(\cdot)$ is a linear functional and can therefore be represented via a Riesz representer. That is, for $\mathbb{P}$-almost every $a \in \mathcal{A}$, there exists a function $\beta_0(a)(\cdot) \in \mathcal{H}_a$ which is the unique Riesz representer satisfying

$$\mathbb{E}[\beta_0(A)(X)\, h(X) \mid A = a] = \mathrm{D}g_{S_0(a)}(h), \tag{11}$$

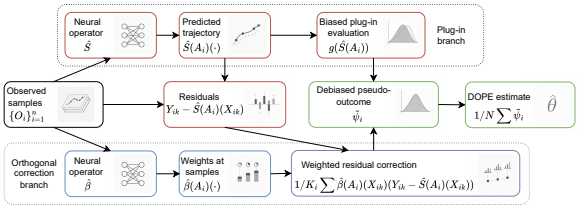

Figure 2: **Overview of DOPE.**

for all $h \in \mathcal{H}_a$. This representation is crucial: it expresses the functional derivative as an expectation *over pointwise evaluations under the sampling design*. Applying Equation 11 with generic $A \in \mathcal{A}$ and $h = \Delta(A)$ shows that the weighted residual $\beta_0(A)(X)\big(Y - \hat{S}(A)(X)\big)$ recovers the leading plug-in bias in expectation. This suggests the *debiasing score*

$$\psi(O) = g(S_0(A)) - \theta + \frac{1}{K}\sum_{k=1}^{K} \beta_0(A)(X_k)\big(Y_k - S_0(A)(X_k)\big), \tag{12}$$

which we formally derive in Lemma B.2. Intuitively, the weighted residual term reconstructs the first-order bias $\mathrm{D}g_{S_0(A)}(\Delta(A))$ and thus provides the required correction. In Supplement B.2, we show that the debiasing score $\psi(O)$ is Neyman-orthogonal at the truth. Motivated by this, we now define DOPE.

Definition 4.2 (DOPE)**.** Given i.i.d. observations $\{O_i\}_{i=1}^n$, consider the debiasing score in Equation 12. We replace the unknown quantities (S_0, β_0) by estimators $(\hat{S}, \hat{\beta})$. We further define the pseudo-outcome

$$\tilde{\psi}_i = g(\hat{S}(A_i)) + \frac{1}{K_i}\sum_{k=1}^{K_i} \hat{\beta}(A_i)(X_{ik})\big(Y_{ik} - \hat{S}(A_i)(X_{ik})\big). \tag{13}$$

In DOPE, the **one-step estimator** of the target functional in Equation 8 is given by

$$\hat{\theta} = \frac{1}{n}\sum_{i=1}^{n} \tilde{\psi}_i. \tag{14}$$

4.3 Theoretical properties

The key property of DOPE is *Neyman-orthogonality*: first-order errors in the neural operator $\hat{S}$ and in the correction term $\hat{\beta}$ do **not** affect the leading asymptotics. As a result, nuisance errors enter the remainder only at *second-order*. We show this formally in Theorem 4.3.

Theorem 4.3 (Second-order remainder)**.** *Let $\hat{\theta}$ be the DOPE estimate in Equation 14. Let $\hat{S}$ and $\hat{\beta}$ be estimated via cross-fitting. Then, we have*

$$\big|\mathbb{E}[\hat{\theta}] - \theta\big| \lesssim \|\hat{S} - S_0\|^2_{L^2(\mathbb{P})} + \|\hat{\beta} - \beta_0\|^2_{L^2(\mathbb{P})}. \tag{15}$$

*In particular, there is **no first-order bias** from either nuisance component.*

Proof. See Supplement B.2. □

As a result, by construction, DOPE *removes the linear bias term* and leaves only a second-order remainder. This is *in contrast* to the plug-in estimator, for which the bias is generally linear in $\mathbb{E}\big[\mathrm{D}g_{S_0(A)}(\Delta(A))\big]$ (cf. Proposition 4.1). Consequently, root-n inference for DOPE is possible under *substantially weaker nuisance rates*. For example, it suffices that

$$\|\hat{S} - S_0\|^2_{L^2(\mathbb{P})} = o_p(n^{-1/2}) \quad \text{and} \quad \|\hat{\beta} - \beta_0\|^2_{L^2(\mathbb{P})} = o_p(n^{-1/2}), \tag{16}$$

which corresponds to $n^{-1/4}$-type rates in $L^2(\mathbb{P})$ root-mean-squared error.

Remark 4.4 (Input-specific targets via second-stage regression)**.** *Our main focus is the population-average estimand $\theta = \mathbb{E}[g(S_0(A))]$. In some applications, however, the object of interest is the input-specific functional*

$$g(S_0(a)) = \mathbb{E}[g(S_0(A)) \mid A = a]. \tag{17}$$

In this case, our debiased pseudo-outcomes $\tilde{\psi}_i$ may be used in a second-stage regression on A_i to estimate the conditional map $a \mapsto g(S_0(a))$. This type of second-stage regression maintains Neyman-orthogonality w.r.t. $\hat{S}$ and $\hat{\beta}$.

Remark 4.5 (Asymptotic normality)**.** *One can further establish asymptotic normality of DOPE. We state a formal result in Supplement B.2, and provide empirical coverage results in Supplement D.*

Remark 4.6 (PPI with neural operators)**.** *Our framework easily integrates prediction-powered inference (PPI) (Angelopoulos et al., 2023a;b), which preserves the othrogonal correction when additional unlabeled inputs are available. Unlike standard PPI, which uses only whether a label is observed, our setting also exploits where labels are observed. Thus, with n_1 partially observed and n_2 fully unlabeled inputs, a PPI-enhanced version of DOPE is*

$$\hat{\theta}_{\mathrm{PPI}} = \frac{1}{n_1+n_2}\sum_{i=1}^{n_1+n_2} g\big(\hat{S}(A_i)\big) + \frac{1}{n_1}\sum_{i=1}^{n_1}\frac{1}{K_i}\sum_{k=1}^{K_i}\hat{\beta}(A_i)(X_{ik})\big(Y_{ik} - \hat{S}(A_i)(X_{ik})\big). \tag{18}$$

4.4 Automatic DML with DOPE

We now turn to the practical question of how to estimate the debiasing weight β_0 from data. The key insight is that β_0 admits a structured decomposition that separates properties of the functional from the sampling design.

Proposition 4.7 (Decomposition of the debiasing weight)**.** *Suppose $\mathcal{U} = L^2(\Omega, \mu)$, and let $w_g(u) \in L^2(\Omega, \mu)$ denote the Riesz representer of $\mathrm{D}g_u$ with respect to μ. Then, the unique solution $\beta_0(A)(x)$ satisfying Equation 11 admits, $\mathbb{P}(\cdot \mid A)$-a.e., the decomposition*

$$\beta_0(A)(x) = \xi_0(A)(x)\, w_g(S_0(A))(x). \tag{19}$$

Proof. See Supplement B.3. □

This proposition shows that the debiasing weight β_0 factorizes into two components: (i) a *functional sensitivity term* w_g, which captures how the target functional reacts to local perturbations of the trajectory, and (ii) a *design correction* ξ_0, which accounts for irregular and input-dependent sampling. Hence, a naïve way to estimate β_0 would be to try to derive w_g analytically, and to obtain ξ_0 by inverting an estimated sampling design $\mathbb{P}(\cdot \mid A)$. However, this requires an inversion of the estimated sampling design, which can be *highly unstable* due to division close to 0.

As a remedy, we show how to estimate the correction term β_0 in the debiasing score *without* explicitly estimating the sampling distribution $\mathbb{P}(\cdot \mid A)$ and performing density inversion. Further, we do not require an analytical derivation of w_g. Instead, our approach directly targets the Riesz representer via a *primal variational characterization*, and thus extends automatic DML to operator-valued nuisances and autodiff-based moments.

Recall from Equation 11 that the correction term β_0 is defined by

$$\mathbb{E}[\beta_0(A)(X)\, h(X) \mid A = a] = \mathrm{D}g_{S_0(a)}(h), \qquad \forall h \in \mathcal{H}_a. \tag{20}$$

Rather than enforcing this equality over a class of test functions h, we exploit the variational characterization of the Riesz representer. For each A, we define the linear functional $\mathcal{L}_A(h) := \mathrm{D}g_{S_0(A)}(h)$. Then, $\beta_0(A)(\cdot)$ is the unique minimizer of the quadratic functional

$$\beta_0(A)(\cdot) = \arg\min_{\beta(A)(\cdot)} \left\{\mathbb{E}[\beta(A)(X)^2 \mid A] - 2\,\mathcal{L}_A\big(\beta(A)(\cdot)\big)\right\}. \tag{21}$$

Averaging over A yields the population risk

$$\beta_0 = \arg\min_{\beta} \mathbb{E}\left[\mathbb{E}[\beta(A)(X)^2 \mid A] - 2\ \mathrm{D}g_{S_0(A)}\big(\beta(A)(\cdot)\big)\right]. \tag{22}$$

Theorem 4.8 (Primal characterization of the Riesz representer)**.** *The unique minimizer of Equation 22 is the Riesz representer β_0 satisfying Equation 20.*

Proof. See Supplement B.3. □

The objective of Equation 22 directly targets the correction weight β_0 required for bias cancellation, without introducing auxiliary test functions h or attempting to estimate the sampling density. In particular, the Riesz identity is enforced implicitly through the optimality conditions of Equation 22, which avoids the need for numerically unstable density inversion.

Remark 4.9 (Structured parameterization when w_g is known)**.** In some cases, the decomposition $\beta_0(A)(x) = \xi_0(A)(x)\, w_g(S_0(A))(x)$ from Equation 19 in Proposition 4.7 admits a closed-form expression for the Riesz representer $w_g(u)$ of the functional derivative $\mathrm{D}g_u$ (see Supplement I). In this case, one may restrict the parameterization of the debiasing weight to $\hat{\beta}(A)(x) = \hat{\xi}(A)(x)\, w_g(\hat{S}(A))(x)$, which reduces the complexity of the estimation problem, and can lead to *improved numerical stability and variance reduction.*

• **Implementation details:** We provide a **pseudo-code** in Algorithm 1 in Supplement E, and we explain how to efficiently compute the empirical analogue of Equation 22 via *automatic differentiation* in Supplement F. Finally, we provide *details on the neural operator backbones* in Supplement G.

5 Experiments

The purpose of our experiments is three-fold: (1) We demonstrate *benefits over simple plug-in estimation*, which is the only existing baseline for estimating functionals with neural operators. (2) We validate the benefits of debiasing, namely, that it offers *robustness to nuisance errors* as derived in Section 4.2. (3) We show that DOPE benefits from large-scale unlabeled data through its *PPI extension*. In Supplement C, we demonstrate the applicability of DOPE on *real-world data*.

• **Implementation:** DOPE is model agnostic and can be instantiated with *any* neural operator architecture. Hence, we thus use a *state-of-the-art backbone*, the Fourier neural operator (Li et al., 2021), in our main results. We provide **additional experiments** with DeepONet (Lu et al., 2021) in Supplement D. We evaluate DOPE against the plug-in estimator, which is the **only** existing baseline for our setting. For fairness reasons, and to isolate the gains from our debiasing, we use the same neural operator backbones for the plug-in baseline and DOPE. Full implementation details are in Supplement G. All results are averaged over 50 runs.

• **Pharmacokinetics dataset:** The pharmacokinetics dataset models the drug concentration under a one-compartment model (Kwon, 2002; Upton et al., 2016). Details are in Supplement H.1.

	$\rho=0$		$\rho=0.125$		$\rho=0.25$		$\rho=0.375$		$\rho=0.5$		$\rho=0.625$		$\rho=0.75$		$\rho=0.875$		$\rho=1$	
	AUC	TAT	AUC	TAT	AUC	TAT	AUC	TAT	AUC	TAT	AUC	TAT	AUC	TAT	AUC	TAT	AUC	TAT
DOPE: structured w_g (ours)	0.72 ± 0.06	0.65 ± 0.06	0.64 ± 0.08	0.59 ± 0.07	0.75 ± 0.06	0.67 ± 0.06	0.89 ± 0.09	0.81 ± 0.07	0.75 ± 0.06	0.67 ± 0.06	0.70 ± 0.09	0.64 ± 0.08	0.75 ± 0.06	0.71 ± 0.06	0.80 ± 0.09	0.70 ± 0.08	0.75 ± 0.07	0.70 ± 0.06
DOPE: oracle β_0 (ours)	0.72 ± 0.06	0.63 ± 0.05	0.65 ± 0.08	0.61 ± 0.08	0.75 ± 0.06	0.65 ± 0.06	0.89 ± 0.09	0.78 ± 0.07	0.75 ± 0.06	0.64 ± 0.05	0.69 ± 0.09	0.62 ± 0.08	0.76 ± 0.06	0.70 ± 0.06	0.80 ± 0.09	0.71 ± 0.08	0.75 ± 0.07	0.68 ± 0.06
Plug-in	1.12 ± 0.09	0.94 ± 0.08	0.92 ± 0.12	0.76 ± 0.09	1.04 ± 0.08	0.83 ± 0.07	1.08 ± 0.10	0.96 ± 0.08	1.06 ± 0.10	0.87 ± 0.08	0.96 ± 0.11	0.80 ± 0.11	0.98 ± 0.09	0.88 ± 0.08	1.08 ± 0.10	0.86 ± 0.08	0.97 ± 0.10	0.86 ± 0.08
DOPE (**ours**)	0.72 ± 0.06	0.65 ± 0.06	0.65 ± 0.08	0.59 ± 0.07	0.75 ± 0.06	0.68 ± 0.06	0.89 ± 0.09	0.80 ± 0.07	0.75 ± 0.06	0.68 ± 0.06	0.70 ± 0.09	0.65 ± 0.09	0.75 ± 0.06	0.71 ± 0.06	0.80 ± 0.09	0.68 ± 0.07	0.75 ± 0.07	0.71 ± 0.06
Rel. improvement	35.4%	30.3%	29.7%	21.9%	28.0%	18.7%	17.3%	17.1%	29.6%	21.3%	27.1%	18.9%	23.6%	19.4%	26.2%	21.1%	22.8%	18.1%

Table 1: **Comparison: plug-in estimator vs. DOPE.** RMSE $\pm$ SE ($\times 100$) across different functionals (area under the concentration curve [AUC] and time-above-threshold [TAT]) for different overlap levels ρ. $\Rightarrow$ *DOPE outperforms the plug-in estimator.*

(1) **Main results (DOPE vs. plug-in):** We analyze 18 settings of different complexity, where we vary (i) the *target functional*: area under the concentration curve (**AUC**) and time-above-threshold (**TAT**) as a more complex functional; (ii) the *sampling irregularity* of the output trajectory for both estimation tasks. For the latter, we vary ρ, where $\rho = 0$ corresponds to uniform sampling, and larger values of ρ increasingly concentrate observations near the subject-specific peak region. We further report a structured parameterization of DOPE with known w_g to show the additional benefits when a closed-form version of the Riesz representer is available. Finally, we report an oracle DOPE where we use the oracle β_0 to understand the upper-bound on the performance.

Results: DOPE outperforms the plug-in baseline by a clear margin (see Table 1). The DOPE variant with structured w_g yields further improvements, which shows that, for known w_g, the structured parametrization simplifies the estimation task notably.

(2) Robustness to nuisance errors: To directly verify the local robustness implied by Neyman orthogonality, we perturb the nuisance functions around their oracle values. Specifically, we perturb the solution operator as $S_\delta = S_0 + \delta h_S$, either alone or jointly with $\beta_\delta = \beta_0 + \delta h_\beta$, for increasing magnitude δ. We then report the resulting absolute bias of the plug-in estimator and DOPE for the smooth TAT functional.

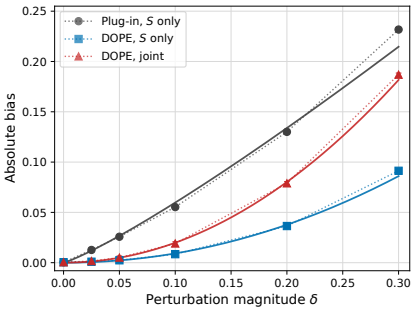

Figure 3: **Oracle-centered nuisance perturbations:** Reported is absolute bias for as we perturb S_0 alone or jointly perturb (S_0, β_0). $\Rightarrow$ *The plug-in bias is first order, while DOPE exhibits only second-order bias.*

Results: Figure 3 shows a clear difference in the local bias order. The plug-in estimator grows approximately linearly with the perturbation magnitude, whereas DOPE exhibits has only quadratic local behavior under both S_0-only and joint (S_0, β_0) perturbations. This is confirmed by the corresponding log–log slopes: 1.16 ± 0.03 for the plug-in estimator, versus 2.03 ± 0.25 (S_0-only) and 2.02 ± 0.12 (joint S_0, β_0) for DOPE, respectively. These results directly confirm our theory: Neyman orthogonality removes the first-order sensitivity to nuisance estimation errors, which leaves only second-order bias.

(3) PPI extension: We report performance of a PPI-style variant of DOPE as in Equation 18, where we increase the number of unlabeled trajectories during inference. We again analyze the performance of DOPE against the plug-in estimator. Specifically, during inference, we increase the number n_2 of input trajectories without any observed point evaluations. Given that DOPE outperforms on AUC and TAT in **(1)–(2)**, we now use the soft continuous maximum (soft cmax) functional (see Supplement I). Results on AUC are in Supplement D.

Results: Table 2 shows that DOPE clearly benefits from additional, unlabeled inputs during inference time, and the gain is larger for DOPE than for the plug-in estimator. This is highly relevant for many scientific scenarios, where generating inputs is cheap, but measuring point evaluations is expensive.

	$n_2 = 0$	$n_2 = 64$	$n_2 = 128$	$n_2 = 256$	$n_2 = 512$	$n_2 = 1024$	$n_2 = 2048$	$n_2 = 4096$
Plug-in	2.90 ± 0.28	2.87 ± 0.27	2.69 ± 0.23	2.48 ± 0.18	2.43 ± 0.21	2.40 ± 0.21	2.35 ± 0.21	2.21 ± 0.19
Rel. improvement	0.0%	0.8%	7.3%	14.4%	16.3%	17.2%	19.0%	23.6%
DOPE	2.75 ± 0.23	2.64 ± 0.19	2.46 ± 0.17	2.27 ± 0.15	2.16 ± 0.13	2.01 ± 0.12	1.87 ± 0.12	1.77 ± 0.11
Rel. improvement	0.0%	4.0%	10.6%	17.5%	21.5%	26.9%	32.0%	35.4%

Table 2: **PPI results.** RMSE $\pm$ SE ($\times 100$), where compare the standard estimates against PPI-style estimates with additional unlabeled samples of size n_2. $\Rightarrow$ *DOPE benefits from access to large unlabeled samples.*

• **Darcy flow dataset:** We consider a two-dimensional Darcy-flow benchmark dataset, which is standard for evaluating neural operator outputs (Kovachki et al., 2021; 2023). Details are in Supplement H.2. We evaluate a smooth excess-above-threshold functional of the Darcy solution field, which measures the spatial extent and intensity of regions where the pressure or state field exceeds a fixed level c. For this, we vary the smoothness parameter κ in the functional.

Results: Table 3 reports the RMSE as we vary the smoothness parameter κ in the target functional. Thereby, we make estimating the target functional more challenging. Due to debiasing, DOPE outperforms the plug-in estimator for all smoothness levels κ.

	$\kappa = 0.0$	$\kappa = 0.2$	$\kappa = 0.4$	$\kappa = 0.6$	$\kappa = 0.8$	$\kappa = 1.0$
Plug-in	0.59 ± 0.06	0.49 ± 0.06	0.41 ± 0.05	0.35 ± 0.04	0.30 ± 0.04	0.26 ± 0.04
DOPE	0.45 ± 0.06	0.39 ± 0.06	0.33 ± 0.05	0.29 ± 0.05	0.26 ± 0.05	0.23 ± 0.05
Rel. improvement	23.5%	21.5%	19.3%	16.8%	14.2%	11.7%

Table 3: **Darcy flow results.** RMSE $\pm$ SE ($\times 10^4$) for plug-in and DOPE across different values of functional smoothness κ. $\Rightarrow$ *DOPE improves over the plug-in estimator across all smoothness levels.*

• **Real-world data:** We report results for the ETTh1 electricity-transformer dataset (Zhou et al., 2021) in Supplement C. Our results confirm that DOPE is applicable to real-world data and consistently outperforms plug-in approaches for different functionals.

• **Conclusion:** In this work, we propose DOPE, a debiased estimator for averaged statistical functionals of neural operator outputs under partial and input-dependent observations. Our approach extends automated DML from function-valued to operator-valued nuisance estimation, and thereby enables bias correction and valid inference for nonlinear functionals of learned solution operators.

AI USE STATEMENT

In this work, we used generative AI tools to assist with implementing the proposed methods, with formulating and proving mathematical claims. We did not use generative AI tools to generate synthetic datasets, develop the theoretical framework, propose or refine hypotheses, design the research methodology or experiments, or interpret experimental results. Translation, dataset cleaning and reformatting, and qualitative or thematic data analysis are not applicable to this work. Additionally, we used generative AI tools to aid paper writing and to cross-check our related-work coverage for potentially overlooked relevant literature; all suggested references were independently verified by the authors. We reviewed all AI-assisted work, including the mathematical arguments and generated experimental artifacts. We take responsibility for the final content of this work, including text, claims, and artifacts produced with the aid of generative AI.

REPRODUCIBILITY STATEMENT

We provide complete proofs for the theoretical results and detailed descriptions of the experimental setup, including the data-generating processes, model configurations, training procedures, evaluation protocols, and random seeds. Code for reproducing the experiments will be made available with the paper. Synthetic datasets are generated from fully specified procedures, and the reported results can be reproduced using the provided implementation.

REFERENCES

Anastasios N Angelopoulos, Stephen Bates, Clara Fannjiang, Michael I Jordan, and Tijana Zrnic. Prediction-powered inference. *Science*, 382(6671):669–674, 2023a.

Anastasios N Angelopoulos, John C Duchi, and Tijana Zrnic. PPI++: Efficient prediction-powered inference. *arXiv preprint arXiv:2311.01453*, 2023b.

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*, 6(5):320–328, 2024.

Jens Berg and Kaj Nyström. Data-driven discovery of PDEs in complex datasets. *Journal of Computational Physics*, 384:239–252, 2019.

Boris Bonev, Thorsten Kurth, Christian Hundt, Jaideep Pathak, Maximilian Baust, Karthik Kashinath, and Anima Anandkumar. Spherical fourier neural operators: Learning stable dynamics on the sphere. In *ICML*, 2023.

Qianying Cao, Somdatta Goswami, and George Em Karniadakis. Laplace neural operator for solving differential equations. *Nature Machine Intelligence*, 6:631–640, 2024.

Marco Carone, Alexander R. Luedtke, and Mark J. van der Laan. Toward computerized efficient estimation in infinite-dimensional models. *Journal of the American Statistical Association*, 114 (527):1174–1190, 2019.

Ricky T. Q. Chen, Yulia Rubanova, Jesse Bettencourt, and David Duvenaud. Neural ordinary differential equations. In *NeurIPS*, 2018.

Victor Chernozhukov, Whitney K. Newey, Victor Quintas-Martinez, and Vasilis Syrgkanis. Automatic debiased machine learning via Riesz regression. *arXiv preprint*, 2104.14737, 2021.

Victor Chernozhukov, Whitney Newey, Victor Quintas-Martinez, and Vasilis Syrgkanis. RieszNet and ForestRiesz: Automatic debiased machine learning with neural nets and random forests. In *ICML*, 2022a.

Victor Chernozhukov, Whitney K. Newey, and Rahul Singh. Automatic debiased machine learning of causal and structural effects. *Econometrica*, 90(3):967–1027, 2022b.

Victor Chernozhukov, Whitney K Newey, and Rahul Singh. Debiased machine learning of global and local parameters using regularized riesz representers. *The Econometrics Journal*, 25(3):576–601, 2022c.

Victor Chernozhukov, Whitney Newey, Victor Quintas-Martinez, and Vasilis Syrgkanis. Automatic debiased machine learning for covariate shifts. *arXiv preprint*, arXiv:2307:04527, 2023.

Alicia Curth and Mihaela van der Schaar. Nonparametric estimation of heterogeneous treatment effects: From theory to learning algorithms. In *AISTATS*, 2021.

Alicia Curth, Ahmed M. Alaa, and Mihaela van der Schaar. Estimating structural target functions using machine learning and influence functions. *arXiv preprint*, arXiv:2008.06461, 2020.

Kangli Dong, Siya Chen, Ying Dan, Lu Zhang, Xinyi Li, Wei Liang, Yue Zhao, and Yu Sun. Optimal stochastic tracking control for brain network dynamics. *Communications Biology*, 8(1), 2025.

Vladimir Fanaskov and Ivan Oseledets. Spectral neural operators. *Doklady Mathematics*, 108(2): 226–232, 2023.

Salah A. Faroughi, Nikhil M. Pawar, Célio Fernandes, Maziar Raissi, Subasish Das, Nima K. Kalantari, and Seyed Kourosh Mahjour. Physics-guided, physics-informed, and physics-encoded neural networks and operators in scientific computing: Fluid and solid mechanics. *Journal of Computing and Information Science in Engineering*, 24(4):040802, 2024.

Dylan J. Foster and Vasilis Syrgkanis. Orthogonal statistical learning. *The Annals of Statistics*, 53(3): 879–908, 2023.

Andrey A. Grachev, Christopher W. Fairall, Byron W. Blomquist, Harindra J.S. Fernando, Laura S. Leo, Sebastián F. Otárola-Bustos, James M. Wilczak, and Katherine L. McCaffrey. On the surface energy balance closure at different temporal scales. *Agricultural and Forest Meteorology*, 281: 107823, 2020.

Yumi Hashiguchi, Naoya Matsumoto, Kazutaka Oda, Hirofumi Jono, and Hideyuki Saito. Population pharmacokinetics and auc-guided dosing of tobramycin in the treatment of infections caused by glucose-nonfermenting gram-negative bacteria. *Clinical Therapeutics*, 45(5):400–414, 2023.

Konstantin Hess and Stefan Feuerriegel. Stabilized neural prediction of potential outcomes in continuous time. In *ICLR*, 2025.

Konstantin Hess, Valentyn Melnychuk, Dennis Frauen, and Stefan Feuerriegel. Bayesian neural controlled differential equations for treatment effect estimation. In *ICLR*, 2024.

Konstantin Hess, Dennis Frauen, Valentyn Melnychuk, and Stefan Feuerriegel. Efficient and sharp off-policy learning under unobserved confounding. In *ICLR*, 2026a.

Konstantin Hess, Dennis Frauen, Mihaela van der Schaar, and Stefan Feuerriegel. Overlap-weighted orthogonal meta-learner for treatment effect estimation over time. In *ICLR*, 2026b.

Oliver Hines, Oliver Dukes, Karla Diaz-Ordaz, and Stijn Vansteelandt. Demystifying statistical learning based on efficient influence functions. *The American Statistician*, 76(3):292–304, 2022.

Valerii Iakovlev, Markus Heinonen, and Harri Lähdesmäki. Learning space-time continuous neural pdes from partially observed states. In *NeurIPS*, 2023.

Hidehiko Ichimura and Whitney K. Newey. The influence function of semiparametric estimators. *Quantitative Economics*, 13(1):29–61, 2022.

Khalid Iqbal, Aliki Milioudi, and Sebastian Georg Wicha. Pharmacokinetics and pharmacodynamics of tedizolid. *Clinical Pharmacokinetics*, 61:489–503, 2022.

Edward H. Kennedy. Semiparametric doubly robust targeted double machine learning: A review. *arXiv preprint*, 2203.06469, 2022.

Yegon Kim, Hyunsu Kim, Gyeonghoon Ko, and Juho Lee. Active learning with selective time-step acquisition for pdes. In *ICML*, 2025.

Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *ICLR*, 2015.

Kazuma Kobayashi, James Daniell, and Syed Bahauddin Alam. Improved generalization with deep neural operators for engineering systems: Path towards digital twin. *Engineering Applications of Artificial Intelligence*, 131:107844, 2024.

Nikola Kovachki, Samuel Lanthaler, and Siddhartha Mishra. On universal approximation and error bounds for fourier neural operators. *Journal of Machine Learning Research*, 22(290):1–76, 2021.

Nikola Kovachki, Zongyi Li, Burigede Liu, Kamyar Azizzadenesheli, Kaushik Bhattacharya, Andrew Stuart, and Anima Anandkumar. Neural operator: Learning maps between function spaces with applications to PDEs. *Journal of Machine Learning Research*, 24(89):1–97, 2023.

Thorsten Kurth, Shashank Subramanian, Peter Harrington, Jaideep Pathak, Morteza Mardani, David Hall, Andrea Miele, Karthik Kashinath, and Anima Anandkumar. FourCastNet: Accelerating global high-resolution weather forecasting using adaptive fourier neural operators. In *PASC*, 2023.

Younggil Kwon. *Handbook of Essential Pharmacokinetics, Pharmacodynamics and Drug Metabolism for Industrial Scientists*. Springer US, Boston, MA, 2002. ISBN 978-0-306-46820-9.

Jin Lee, Ziming Liu, Xinling Yu, Yixuan Wang, Haewon Jeong, Murphy Yuezhen Niu, and Zheng Zhang. KANO: Kolmogorov-Arnold neural operator. In *ICLR*, 2026.

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

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

Alex Luedtke. Simplifying debiased inference via automatic differentiation and probabilistic programming. *arXiv preprint*, 2405.08675, 2024.

Valentyn Melnychuk, Stefan Feuerriegel, and Mihaela van der Schaar. Quantifying aleatoric uncertainty of the treatment effect: A novel orthogonal learner. In *NeurIPS*, 2024.

Lucas Meyer, Marc Schouler, Robert A. Caulk, Alejandro Ribes, and Bruno Raffin. Training deep surrogate models with large scale online learning. In *ICML*, 2023.

Pawel Morzywolek, Johan Decruyenaere, and Stijn Vansteelandt. On a general class of orthogonal learners for the estimation of heterogeneous treatment effects. *arXiv preprint*, arXiv:2303.12687, 2023.

Marco Pavone. On the riesz representation theorem for bounded linear functionals. *Proceedings of the Royal Irish Academy. Section A: Mathematical and Physical Sciences*, 94A(1):133–135, 1994.

Wenhui Peng, Shaoxiang Qin, Senwen Yang, Jianchun Wang, Xue Liu, and Liangzhu (Leon) Wang. Fourier neural operator for real-time simulation of 3d dynamic urban microclimate. *Building and Environment*, 248:111063, 2024.

Ilan Price, Alvaro Sanchez-Gonzalez, Ferran Alet, Tom R. Andersson, Andrew El-Kadi, Dominic Masters, Timo Ewalds, Jacklynn Stott, Shakir Mohamed, Peter Battaglia, Remi Lam, and Matthew Willson. Probabilistic weather forecasting with machine learning. *Nature*, 637:84–90, 2025.

Md Ashiqur Rahman, Zachary E. Ross, and Kamyar Azizzadenesheli. U-NO: U-shaped neural operators. *Transactions on Machine Learning Research*, 2023.

Bogdan Raonic, Roberto Molinaro, Tim De Ryck, Tobias Rohner, Francesca Bartolucci, Siddhartha Alaifari, Rima Mishra, and Emmanuel de Bézenac. Convolutional neural operators for robust and accurate learning of pdes. In *NeurIPS*, 2023.

Shawn G. Rosofsky, Hani Al Majed, and E A Huerta. Applications of physics informed neural operators. *Machine Learning: Science and Technology*, 4, 2023.

Khemraj Shukla, Vivek Oommen, Ahmad Peyvan, Michael Penwarden, Nicholas Plewacki, Luis Bravo, Anindya Ghoshal, Robert M. Kirby, and George Em Karniadakis. Deep neural operators as accurate surrogates for shape optimization. *Engineering Applications of Artificial Intelligence*, 129:107615, 2024.

Richard N. Upton, David J.R. Foster, and Ahmad Y. Abuhelwa. An introduction to physiologically-based pharmacokinetic models. *Pediatric Anesthesia*, 26(11):1036–1046, 2016.

Lars van der Laan, Aurelien Bibaut, Nathan Kallus, and Alex Luedtke. Automatic debiased machine learning for smooth functionals of nonparametric M-estimands. *arXiv preprint*, arXiv:2501.11868, 2025.

Hui Wang, Yang Song, Haonan Yang, and Zhigang Liu. Generalized koopman neural operator for data-driven modeling of electric railway pantograph–catenary systems. *IEEE Transactions on Transportation Electrification*, 11(6):14100–14112, 2025.

Gege Wen, Zongyi Li, Qirui Long, Kamyar Azizzadenesheli, Anima Anandkumar, and Sally M. Benson. Real-time high-resolution co2 geological storage prediction using nested fourier neural operators. *Science*, 16:1732–1741, 2023.

Afton Widdershins, Elsa Hansen, Andrew Read, and Raymond Hohl. Optimal control theory as a method for designing multidrug adaptive therapy regimens. *npj Systems Biology and Applications*, 12(1), 2026.

Xi-Ding Yang and Yong-Yu Yang. Clinical pharmacokinetics of semaglutide: A systematic review. *Drug Design, Development and Therapy*, 18:2555–2570, 2024.

Zhiwei Zhao, Changqing Liu, Yingguang Li, Zhibin Chen, and Xu Liu. Diffeomorphism neural operator for various domains and parameters of partial differential equations. *Communications Physics*, 8(15), 2025.

Haoyi Zhou, Shanghang Zhang, Jieqi Peng, Shuai Zhang, Jianxin Li, Hui Xiong, and Wancai Zhang. Informer: Beyond efficient transformer for long sequence time-series forecasting. In *AAAI*, 2021.

Yue Zhou, Junhao Li, Zhongyi Xu, Yixin Zhao, Shanshan Zhang, Tong Liu, Yelan Yao, Lu Fang, Yu Cai, , Xuesong Ye, and Bo Liang. A nanomip sensor for real-time in vivo monitoring of levodopa pharmacokinetics in precision parkinson's therapy. *Nature Communications*, 16(1), 2025.

A EXTENDED RELATED WORK

Neural operators provide a framework for learning mappings between function spaces (Kovachki et al., 2023). This allows one to learn solution operators for PDEs that map input functions (e.g., coefficients or boundary conditions) to solution trajectories. Hence, neural operators are widely used to solve scientific problems in scientific computing (Azizzadenesheli et al., 2024; Cao et al., 2024; Faroughi et al., 2024), climate science (Kurth et al., 2023; Peng et al., 2024; Wen et al., 2023), and engineering (Faroughi et al., 2024; Kobayashi et al., 2024; Shukla et al., 2024; Wang et al., 2025). They enjoy several favorable properties, such as discretization invariance, the ability to evaluate functions at arbitrary locations, and universal approximation properties (Kovachki et al., 2021; 2023).

Several architectures for neural networks have been proposed (e.g., Bonev et al., 2023; Cao et al., 2024; Fanaskov & Oseledets, 2023; Lee et al., 2026; Lu et al., 2021; Rahman et al., 2023; Raonic et al., 2023; Zhao et al., 2025). Two state-of-the-art examples are the Fourier neural operator (Li et al., 2021) and DeepONet (Lu et al., 2021), which we use both to demonstrate our framework. Importantly, our DOPE framework is agnostic to the underlying architecture and can be instantiated with *arbitrary* neural operator backbones.

A common downstream use of neural operators is inference on low-dimensional functionals of the predicted trajectory rather than recovery of the full field itself (see Figure 1). $\Rightarrow$ *However, the only generally available strategy is naïve plug-in evaluation of the functional on the predicted trajectory, which suffers from* ***plug-in bias*** *under operator approximation error.*

B PROOFS

B.1 PLUG-IN BIAS

Proposition B.1 (First-order bias of plug-in estimators)**.** *Let* $\Delta(\cdot) = \hat{S}(\cdot) - S_0(\cdot)$. *Then*

$$\left|\mathbb{E}[g(\hat{S}(A))] - \theta\right| \lesssim \left|\mathbb{E}\big[\mathrm{D}g_{S_0(A)}(\Delta(A))\big]\right| + \|\Delta\|^2_{L^2(\mathbb{P})}, \tag{23}$$

i.e., the neural operator error propagates linearly through the Fréchet differential and induces first-order bias.

Proof. We apply the second-order expansion in Equation 7 with $h = \Delta(A)$:

$$g(\hat{S}(A)) = g(S_0(A) + \Delta(A)) = g(S_0(A)) + \mathrm{D}g_{S_0(A)}(\Delta(A)) + R(S_0(A), \Delta(A)). \tag{24}$$

Taking expectations and using $\theta = \mathbb{E}[g(S_0(A))]$ yields

$$\mathbb{E}[g(\hat{S}(A))] - \theta = \mathbb{E}\big[\mathrm{D}g_{S_0(A)}(\Delta(A))\big] + \mathbb{E}[R(S_0(A), \Delta(A))]. \tag{25}$$

In the Hilbert space $\mathcal{U} = L^2(\Omega, \mu)$, the remainder bound in Equation 7 gives

$$|R(S_0(A), \Delta(A))| \lesssim \|\Delta(A)\|^2_{L^2(\mu)}, \tag{26}$$

which depends on A only. In order to express this bound in the sampling geometry, we use Assumption 3.1. That is, since $\mu \ll \mathbb{P}(\cdot \mid A)$ with Radon–Nikodym derivative $\xi_0(A)(\cdot) = \frac{\mathrm{d}\mu}{\mathrm{d}\mathbb{P}(\cdot|A)} \leq C$, we have, for each realization of A,

$$\|\Delta(A)\|^2_{L^2(\mu)} = \int_\Omega \Delta(A)(x)^2 \, \mathrm{d}\mu(x) = \int_\Omega \Delta(A)(x)^2 \, \xi_0(A)(x) \, \mathrm{d}\mathbb{P}(x \mid A) \tag{27}$$

$$\leq C \int_\Omega \Delta(A)(x)^2 \, \mathrm{d}\mathbb{P}(x \mid A) = C\mathbb{E}[\Delta(A)(X)^2 \mid A]. \tag{28}$$

Taking expectation over A and using the definition of the joint law $\mathbb{P}(\mathrm{d}a, \mathrm{d}x) = \mathbb{P}(\mathrm{d}a)\,\mathbb{P}(\mathrm{d}x \mid a)$ yields

$$\mathbb{E}\|\Delta(A)\|^2_{L^2(\mu)} \leq C\,\mathbb{E}[\Delta(A)(X)^2] = C\,\|\Delta\|^2_{L^2(\mathbb{P})}, \qquad \|\Delta\|^2_{L^2(\mathbb{P})} \coloneqq \mathbb{E}[\Delta(A)(X)^2]. \tag{29}$$

Therefore, applying Jensen's inequality and Equation 29, yields

$$\left|\mathbb{E}[R(S_0(A), \Delta(A))]\right| \leq \mathbb{E}\big[|R(S_0(A), \Delta(A))|\big] \lesssim \mathbb{E}\|\Delta(A)\|^2_{L^2(\mu)} \lesssim \|\Delta\|^2_{L^2(\mathbb{P})}. \tag{30}$$

Finally, combining Equation 25 and Equation 30 and applying the triangle inequality gives

$$\left|\mathbb{E}[g(\hat{S}(A))] - \theta\right| \lesssim \left|\mathbb{E}\big[\mathrm{D}g_{S_0(A)}(\Delta(A))\big]\right| + \|\Delta\|^2_{L^2(\mathbb{P})}. \tag{31}$$

Unless the linear term vanishes (which it need not for nonlinear g), $\sqrt{n}$-inference requires this bias to be $o(n^{-1/2})$, i.e.,

$$\mathbb{E}\big[\mathrm{D}g_{S_0(A)}(\Delta(A))\big] = o(n^{-1/2}), \tag{32}$$

which is a first-order (linear) accuracy requirement on the operator error $\Delta(A)$.

□

B.2 ONE-STEP ESTIMATOR

Lemma B.2 (Debiasing identity for the score)**.** *Let Δ be any measurable perturbation such that $\Delta(A,\cdot) \in \mathcal{U}$ almost surely and*

$$\mathbb{E}[\Delta(A)(X)^2] < \infty. \tag{33}$$

If $\beta_0 \in L^2(\mathbb{P})$ satisfies

$$\mathbb{E}[\beta_0(A)(X)h(X) \mid A] = \mathrm{D}g_{S_0(A)}(h), \qquad \forall h \in \mathcal{H}_A, \tag{34}$$

then

$$\mathbb{E}\left[\frac{1}{K}\sum_{k=1}^{K}\beta_0(A)(X_k)\Delta(A)(X_k)\right] = \mathbb{E}\big[\mathrm{D}g_{S_0(A)}(\Delta(A))\big]\,. \tag{35}$$

Moreover, if $\hat{S}$ is any estimator independent of the evaluation fold (for instance, by cross-fitting), and $\Delta(A) = \hat{S}(A) - S_0(A)$, then

$$\mathbb{E}\left[\frac{1}{K}\sum_{k=1}^{K}\beta_0(A)(X_k)\big(Y_k - \hat{S}(A)(X_k)\big)\,\middle|\,\hat{S}\right] = -\mathbb{E}\Big[\mathrm{D}g_{S_0(A)}(\Delta(A))\,\Big|\,\hat{S}\Big]\,. \tag{36}$$

Hence, the weighted residual term in Equation 12 cancels the linear plug-in bias.

Proof. For the first claim, conditional on (A, K) the pairs $\{(X_k, Y_k)\}_{k=1}^{K}$ are i.i.d., so applying exchangeability yields

$$\mathbb{E}\left[\frac{1}{K}\sum_{k=1}^{K}\beta_0(A)(X_k)\Delta(A)(X_k)\,\middle|\,A,K\right] = \mathbb{E}[\beta_0(A)(X)\Delta(A)(X) \mid A]. \tag{37}$$

Taking expectations and applying the Riesz identity from Equation 34 with $h = \Delta(A)$ yields

$$\mathbb{E}\left[\frac{1}{K}\sum_{k=1}^{K}\beta_0(A)(X_k)\Delta(A)(X_k)\right] = \mathbb{E}[\mathbb{E}[\beta_0(A)(X)\Delta(A)(X) \mid A]] \tag{38}$$

$$= \mathbb{E}\big[\mathrm{D}g_{S_0(A)}(\Delta(A))\big]\,, \tag{39}$$

which proves Equation 35.

For the second claim, write

$$Y_k - \hat{S}(A)(X_k) = \big(Y_k - S_0(A)(X_k)\big) - \Delta(A)(X_k), \qquad \Delta(A) = \hat{S}(A) - S_0(A). \tag{40}$$

Conditioning on $\hat{S}$ treats Δ as fixed, so

$$\begin{aligned}&\mathbb{E}\left[\frac{1}{K}\sum_{k=1}^{K}\beta_0(A)(X_k)\big(Y_k - \hat{S}(A)(X_k)\big)\,\middle|\,\hat{S}\right]\\ &= \mathbb{E}\left[\frac{1}{K}\sum_{k=1}^{K}\beta_0(A)(X_k)\big(Y_k - S_0(A)(X_k)\big)\,\middle|\,\hat{S}\right] - \mathbb{E}\left[\frac{1}{K}\sum_{k=1}^{K}\beta_0(A)(X_k)\Delta(A)(X_k)\,\middle|\,\hat{S}\right].\end{aligned} \tag{41}$$

For the first term, iterated expectations together with $\mathbb{E}[Y_k - S_0(A)(X_k) \mid A, X_k] = 0$ give

$$\mathbb{E}\left[\frac{1}{K}\sum_{k=1}^{K}\beta_0(A)(X_k)\big(Y_k - S_0(A)(X_k)\big)\,\middle|\,\hat{S}\right] = 0. \tag{42}$$

Applying Equation 35 conditionally on $\hat{S}$ to the second term in Equation 41 yields

$$\mathbb{E}\left[\frac{1}{K}\sum_{k=1}^{K}\beta_0(A)(X_k)\big(Y_k - \hat{S}(A)(X_k)\big)\,\middle|\,\hat{S}\right] = -\mathbb{E}\Big[\mathrm{D}g_{S_0(A)}(\Delta(A))\,\Big|\,\hat{S}\Big]\,, \tag{43}$$

which proves Equation 36. □

Proposition B.3 (Neyman orthogonality of the debiasing score)**.** *Define the score map*

$$m(O;\theta,S,\beta) \coloneqq g(S(A)) - \theta + \frac{1}{K}\sum_{k=1}^{K} \beta(A)(X_k)\big(Y_k - S(A)(X_k)\big). \tag{44}$$

Let (θ_0, S_0, β_0) *denote the truth, where*

$$\theta_0 = \mathbb{E}[g(S_0(A))] \tag{45}$$

and $\beta_0 \in L^2(\mathbb{P})$ *satisfies*

$$\mathbb{E}[\beta_0(A)(X)h(X) \mid A] = \mathrm{D}g_{S_0(A)}(h) \qquad \forall h \in \mathcal{H}_A. \tag{46}$$

Then, the score is Neyman-orthogonal at (θ_0, S_0, β_0) *in the sense that, for any admissible perturbations* h *and* b *such that* $h(A,\cdot) \in \mathcal{H}_A$ *almost surely,* $b \in L^2(\mathbb{P})$*, and* $S_0 + th$ *remains in the neighborhood from Equation 7 for sufficiently small* t*, i.e.,*

$$\frac{\mathrm{d}}{\mathrm{d}t}\mathbb{E}[m(O;\theta_0, S_0 + th, \beta_0)]\Big|_{t=0} = 0, \tag{47}$$

$$\frac{\mathrm{d}}{\mathrm{d}t}\mathbb{E}[m(O;\theta_0, S_0, \beta_0 + tb)]\Big|_{t=0} = 0. \tag{48}$$

Proof. We first consider perturbations of S. By definition of the score map,

$$\mathbb{E}[m(O;\theta_0, S_0 + th, \beta_0)] = \mathbb{E}[g(S_0(A) + th(A))] - \theta_0 \tag{49}$$

$$+ \mathbb{E}\left[\frac{1}{K}\sum_{k=1}^{K} \beta_0(A)(X_k)\big(Y_k - S_0(A)(X_k) - th(A)(X_k)\big)\right]. \tag{50}$$

Differentiating at $t = 0$ using Fréchet differentiability of g gives

$$\frac{\mathrm{d}}{\mathrm{d}t}\mathbb{E}[m(O;\theta_0, S_0 + th, \beta_0)]\Big|_{t=0} = \mathbb{E}[\mathrm{D}g_{S_0(A)}(h(A))] - \mathbb{E}\left[\frac{1}{K}\sum_{k=1}^{K} \beta_0(A)(X_k)h(A)(X_k)\right]. \tag{51}$$

Conditional on (A, K), the pairs $\{(X_k, Y_k)\}_{k=1}^{K}$ are i.i.d., so by exchangeability, we arrive at

$$\mathbb{E}\left[\frac{1}{K}\sum_{k=1}^{K} \beta_0(A)(X_k)h(A)(X_k)\,\middle|\, A, K\right] = \mathbb{E}[\beta_0(A)(X)h(A)(X) \mid A]. \tag{52}$$

Hence, we obtain

$$\mathbb{E}\left[\frac{1}{K}\sum_{k=1}^{K} \beta_0(A)(X_k)h(A)(X_k)\right] = \mathbb{E}[\mathbb{E}[\beta_0(A)(X)h(A)(X) \mid A]] \tag{53}$$

$$= \mathbb{E}[\mathrm{D}g_{S_0(A)}(h(A))], \tag{54}$$

where the last step uses the Riesz identity from Equation 46 with $h = h(A,\cdot)$. Substituting into Equation 51 proves Equation 47.

Next, consider perturbations of β. We have

$$\mathbb{E}[m(O;\theta_0, S_0, \beta_0 + tb)] = \mathbb{E}[m(O;\theta_0, S_0, \beta_0)] + t\,\mathbb{E}\left[\frac{1}{K}\sum_{k=1}^{K} b(A, X_k)\big(Y_k - S_0(A)(X_k)\big)\right]. \tag{55}$$

Therefore,

$$\frac{\mathrm{d}}{\mathrm{d}t}\mathbb{E}[m(O;\theta_0, S_0, \beta_0 + tb)]\Big|_{t=0} = \mathbb{E}\left[\frac{1}{K}\sum_{k=1}^{K} b(A, X_k)\big(Y_k - S_0(A)(X_k)\big)\right]. \tag{56}$$

Using iterated expectations and the regression property $\mathbb{E}[Y_k - S_0(A)(X_k) \mid A, X_k] = 0$, we obtain

$$\mathbb{E}\big[b(A, X_k)\big(Y_k - S_0(A)(X_k)\big)\big] = \mathbb{E}[b(A, X_k)\,\mathbb{E}[Y_k - S_0(A)(X_k) \mid A, X_k]] = 0. \tag{57}$$

Averaging over k yields Equation 48. Thus, the score is Neyman-orthogonal at (θ_0, S_0, β_0). □

Theorem B.4 (Second-order remainder)**.** *Let $\hat{\theta}$ be the one-step estimator in Equation 14. If $\hat{S}$ and $\hat{\beta}$ are cross-fitted, then*

$$\left|\mathbb{E}[\hat{\theta}] - \theta\right| \lesssim \|\hat{S} - S_0\|^2_{L^2(\mathbb{P})} + \|\hat{\beta} - \beta_0\|^2_{L^2(\mathbb{P})}. \tag{58}$$

*In particular, there is **no first-order bias** in either nuisance component.*

Proof. Recall that

$$\hat{\theta} = \frac{1}{n}\sum_{i=1}^{n} \tilde{\psi}_i, \qquad \tilde{\psi} = g(\hat{S}(A)) + \frac{1}{K}\sum_{k=1}^{K} \hat{\beta}(A)(X_k)\big(Y_k - \hat{S}(A)(X_k)\big), \tag{59}$$

where $(A, K, \{(X_k, Y_k)\}_{k=1}^K)$ denotes a generic observation in the evaluation fold.

Step 0: Cross-fitting. By cross-fitting, $(\hat{S}, \hat{\beta})$ are measurable with respect to the training sample $\mathcal{D}^{\text{train}}$, and independent of the evaluation fold. Thus,

$$\mathbb{E}[\hat{\theta} - \theta] = \mathbb{E}\Big[\mathbb{E}[\tilde{\psi} - \theta \mid \mathcal{D}^{\text{train}}]\Big], \tag{60}$$

and it therefore suffices to bound the conditional bias $\mathbb{E}[\tilde{\psi} - \theta \mid \mathcal{D}^{\text{train}}]$.

Step 1: Expansion of functional. Let $\Delta(A) = \hat{S}(A) - S_0(A)$. By Equation 7, we have that

$$g(\hat{S}(A)) = g(S_0(A)) + \mathrm{D}g_{S_0(A)}(\Delta(A)) + R(S_0(A), \Delta(A)), \qquad |R(S_0(A), \Delta(A))| \lesssim \|\Delta(A)\|^2_{\mathcal{U}}. \tag{61}$$

Hence, taking conditional expectation given $\mathcal{D}^{\text{train}}$ yields

$$\mathbb{E}[g(\hat{S}(A)) - g(S_0(A)) \mid \mathcal{D}^{\text{train}}] = \mathbb{E}[\mathrm{D}g_{S_0(A)}(\Delta(A)) \mid \mathcal{D}^{\text{train}}] + O\big(\mathbb{E}[\|\Delta(A)\|^2_{\mathcal{U}} \mid \mathcal{D}^{\text{train}}]\big). \tag{62}$$

Step 2: Expansion of the residual correction. Next, we decompose

$$Y_k - \hat{S}(A)(X_k) = (Y_k - S_0(A)(X_k)) - \Delta(A)(X_k) =: \varepsilon_k - \Delta(A)(X_k). \tag{63}$$

Since $\mathbb{E}[\varepsilon_k \mid A, X_k] = 0$ and $\hat{\beta}$ is measurable w.r.t. $\mathcal{D}^{\text{train}}$, iterated expectations give

$$\mathbb{E}[\hat{\beta}(A)(X_k)\varepsilon_k \mid \mathcal{D}^{\text{train}}] = 0 \tag{64}$$

Therefore, we have that

$$\mathbb{E}\left[\frac{1}{K}\sum_{k=1}^{K} \hat{\beta}(A)(X_k)\big(Y_k - \hat{S}(A)(X_k)\big) \,\middle|\, \mathcal{D}^{\text{train}}\right] = -\mathbb{E}[\hat{\beta}(A)(X)\Delta(A)(X) \mid \mathcal{D}^{\text{train}}], \tag{65}$$

where $X \mid A \sim \mathbb{P}(\cdot \mid A)$ denotes a generic draw from the sampling design.

With $\hat{\beta} = \beta_0 + (\hat{\beta} - \beta_0)$, we have

$$-\mathbb{E}[\hat{\beta}(A)(X)\Delta(A)(X) \mid \mathcal{D}^{\text{train}}] \tag{66}$$

$$= -\mathbb{E}[\beta_0(A)(X)\Delta(A)(X) \mid \mathcal{D}^{\text{train}}] - \mathbb{E}[(\hat{\beta} - \beta_0)(A)(X)\Delta(A)(X) \mid \mathcal{D}^{\text{train}}]. \tag{67}$$

Step 3: First-order cancellation. By the Riesz identity in Equation 11, we obtain

$$\mathbb{E}[\beta_0(A)(X)\Delta(A)(X) \mid A] = \mathrm{D}g_{S_0(A)}(\Delta(A)). \tag{68}$$

Hence, we take expectations conditionally on $\mathcal{D}^{\text{train}}$, which yields

$$\mathbb{E}[\beta_0(A)(X)\Delta(A)(X) \mid \mathcal{D}^{\text{train}}] = \mathbb{E}[\mathrm{D}g_{S_0(A)}(\Delta(A)) \mid \mathcal{D}^{\text{train}}]. \tag{69}$$

Substituting Equation 69 into Equation 67 and Equation 65, we obtain

$$\mathbb{E}\left[\frac{1}{K}\sum_{k=1}^{K}\hat{\beta}(A,X_k)\big(Y_k-\hat{S}(A)(X_k)\big)\,\middle|\,\mathcal{D}^{\text{train}}\right]=-\mathbb{E}[\mathrm{D}g_{S_0(A)}(\Delta(A))\mid\mathcal{D}^{\text{train}}]$$
$$-\mathbb{E}[(\hat{\beta}-\beta_0)(A,X)\Delta(A)(X)\mid\mathcal{D}^{\text{train}}]. \quad (70)$$

Hence, the β_0-part of the correction term exactly cancels the first-order term

$$\mathbb{E}[\mathrm{D}g_{S_0(A)}(\Delta(A))\mid\mathcal{D}^{\text{train}}] \quad (71)$$

from the Taylor expansion in Equation 62, while the additional cross-term involving $(\hat{\beta}-\beta_0)\Delta$ remains and is controlled in the next step.

Step 4: Bounding the remaining second-order terms. Combining Equation 62 with Equation 70, after cancellation of the linear term, yields

$$\mathbb{E}[\tilde{\psi}-\theta\mid\mathcal{D}^{\text{train}}]=O\big(\mathbb{E}[\|\Delta(A)\|_{\mathcal{U}}^2\mid\mathcal{D}^{\text{train}}]\big)-\mathbb{E}[(\hat{\beta}-\beta_0)(A,X)\Delta(A)(X)\mid\mathcal{D}^{\text{train}}]. \quad (72)$$

For the cross-term, conditional Cauchy–Schwarz gives

$$\Big|\mathbb{E}[(\hat{\beta}-\beta_0)(A)(X)\Delta(A)(X)\mid\mathcal{D}^{\text{train}}]\Big|\le\|\hat{\beta}-\beta_0\|_{L^2(\mathbb{P}|\mathcal{D}^{\text{train}})}\cdot\|\Delta(A)(X)\|_{L^2(\mathbb{P}|\mathcal{D}^{\text{train}})}. \quad (73)$$

Hence, using $2ab\le a^2+b^2$ yields

$$\Big|\mathbb{E}[(\hat{\beta}-\beta_0)(A)(X)\Delta(A)(X)\mid\mathcal{D}^{\text{train}}]\Big|\le\frac{1}{2}\|\hat{\beta}-\beta_0\|^2_{L^2(\mathbb{P}|\mathcal{D}^{\text{train}})}+\frac{1}{2}\,\mathbb{E}[\Delta(A)(X)^2\mid\mathcal{D}^{\text{train}}]. \quad (74)$$

Finally, by Assumption 3.1 (i.e., bounded $\xi_0(A)=\mathrm{d}\mu/\,\mathrm{d}\mathbb{P}(\cdot\mid A)$) and $\mathcal{U}=L^2(\Omega,\mu)$,

$$\mathbb{E}[\|\Delta(A)\|_{\mathcal{U}}^2\mid\mathcal{D}^{\text{train}}]=\mathbb{E}\left[\int_\Omega\Delta(A)(x)^2\,\mathrm{d}\mu(x)\,\middle|\,\mathcal{D}^{\text{train}}\right] \quad (75)$$
$$\le C\,\mathbb{E}\left[\int_\Omega\Delta(A)(x)^2\,\mathrm{d}\mathbb{P}(x\mid A)\,\middle|\,\mathcal{D}^{\text{train}}\right]=C\,\mathbb{E}[\Delta(A)(X)^2\mid\mathcal{D}^{\text{train}}]. \quad (76)$$

Substituting Equation 74 and Equation 76 into Equation 72 then gives

$$\big|\mathbb{E}[\tilde{\psi}-\theta\mid\mathcal{D}^{\text{train}}]\big|\ \lesssim\ \|\hat{\beta}-\beta_0\|^2_{L^2(\mathbb{P}|\mathcal{D}^{\text{train}})}+\mathbb{E}[\Delta(A)(X)^2\mid\mathcal{D}^{\text{train}}]. \quad (77)$$

Finally, taking expectation over $\mathcal{D}^{\text{train}}$ together with the tower property yields

$$\big|\mathbb{E}[\hat{\theta}]-\theta\big|\ \lesssim\ \|\hat{\beta}-\beta_0\|^2_{L^2(\mathbb{P})}+\mathbb{E}[\Delta(A)(X)^2]=\|\hat{\beta}-\beta_0\|^2_{L^2(\mathbb{P})}+\|\hat{S}-S_0\|^2_{L^2(\mathbb{P})}, \quad (78)$$

which concludes the proof. □

Proposition B.5 (Asymptotic normality). *If $\hat{S}$ and $\hat{\beta}$ are cross-fitted, $\mathbb{E}[\psi_i^2] < \infty$, and the nuisance estimators satisfy*

$$\|\hat{S} - S_0\|_{L^2(\mathbb{P})}^2 = o_p(n^{-1/2}), \qquad \|\hat{\beta} - \beta_0\|_{L^2(\mathbb{P})}^2 = o_p(n^{-1/2}), \tag{79}$$

and, in addition, the estimated score is conditionally mean-square consistent:

$$\mathbb{E}\Big[(\tilde{\psi}_i - \psi_i)^2 \mid \mathcal{D}^{\text{train}}\Big] = o_p(1), \tag{80}$$

then

$$\sqrt{n}(\hat{\theta} - \theta) \xrightarrow{d} \mathcal{N}\big(0, \mathbb{V}[\psi_i]\big). \tag{81}$$

Moreover, a consistent estimator of the asymptotic variance of $\hat{\theta}$ is

$$\widehat{\mathbb{V}}(\hat{\theta}) = \frac{1}{n^2} \sum_{i=1}^{n} (\tilde{\psi}_i - \hat{\theta})^2. \tag{82}$$

Proof. Let

$$\delta_i \coloneqq \tilde{\psi}_i - \psi_i. \tag{83}$$

Then

$$\hat{\theta} - \theta = \frac{1}{n} \sum_{i=1}^{n} \psi_i + \frac{1}{n} \sum_{i=1}^{n} \delta_i. \tag{84}$$

Since $\mathbb{E}[\psi_i] = 0$, we may further decompose

$$\sqrt{n}(\hat{\theta} - \theta) = \frac{1}{\sqrt{n}} \sum_{i=1}^{n} \psi_i + \frac{1}{\sqrt{n}} \sum_{i=1}^{n} \Big(\delta_i - \mathbb{E}[\delta_i \mid \mathcal{D}^{\text{train}}]\Big) + \sqrt{n}\, \mathbb{E}[\delta_i \mid \mathcal{D}^{\text{train}}]. \tag{85}$$

Step 1: Bias term. By definition, $\delta_i = \tilde{\psi}_i - \psi_i$, and since $\mathbb{E}[\psi_i] = 0$, we have

$$\mathbb{E}[\delta_i \mid \mathcal{D}^{\text{train}}] = \mathbb{E}[\tilde{\psi}_i - \theta \mid \mathcal{D}^{\text{train}}]. \tag{86}$$

Hence, by the second-order remainder result (Theorem B.4), we obtain

$$\mathbb{E}[\delta_i \mid \mathcal{D}^{\text{train}}] = O_p\Big(\|\hat{S} - S_0\|_{L^2(\mathbb{P})}^2 + \|\hat{\beta} - \beta_0\|_{L^2(\mathbb{P})}^2\Big). \tag{87}$$

Under Equation 79, this yields

$$\sqrt{n}\, \mathbb{E}[\delta_i \mid \mathcal{D}^{\text{train}}] = o_p(1). \tag{88}$$

Step 2: Stochastic remainder. Conditional on $\mathcal{D}^{\text{train}}$, the variables $\{\delta_i\}_{i=1}^n$ are i.i.d. Therefore,

$$\mathbb{V}\left(\frac{1}{\sqrt{n}} \sum_{i=1}^{n} \big(\delta_i - \mathbb{E}[\delta_i \mid \mathcal{D}^{\text{train}}]\big) \,\middle|\, \mathcal{D}^{\text{train}}\right) = \mathbb{V}(\delta_i \mid \mathcal{D}^{\text{train}}) \le \mathbb{E}[\delta_i^2 \mid \mathcal{D}^{\text{train}}]. \tag{89}$$

By Equation 80, the right-hand side is $o_p(1)$, so conditional Chebyshev implies

$$\frac{1}{\sqrt{n}} \sum_{i=1}^{n} \big(\delta_i - \mathbb{E}[\delta_i \mid \mathcal{D}^{\text{train}}]\big) = o_p(1). \tag{90}$$

Step 3: Central limit theorem (CLT). Combining Equation 85, Equation 88, and Equation 90, we obtain

$$\sqrt{n}(\hat{\theta} - \theta) = \frac{1}{\sqrt{n}} \sum_{i=1}^{n} \psi_i + o_p(1). \tag{91}$$

Since $\{\psi_i\}_{i=1}^n$ are i.i.d., mean zero, and have finite variance, the classical CLT gives

$$\frac{1}{\sqrt{n}} \sum_{i=1}^{n} \psi_i \xrightarrow{d} \mathcal{N}\big(0, \mathbb{V}[\psi_i]\big). \tag{92}$$

Slutsky's theorem then yields

$$\sqrt{n}(\hat{\theta} - \theta) \xrightarrow{d} \mathcal{N}\big(0, \mathbb{V}[\psi_i]\big). \tag{93}$$

Step 4: Variance estimation. We first show that

$$\frac{1}{n} \sum_{i=1}^{n} (\tilde{\psi}_i - \hat{\theta})^2 \xrightarrow{p} \mathbb{V}[\psi_i]. \tag{94}$$

Write again $\tilde{\psi}_i = \psi_i + \delta_i$. Then, conditional on $\mathcal{D}^{\text{train}}$, the variables δ_i are i.i.d., so by the weak law of large numbers and Equation 80,

$$\frac{1}{n} \sum_{i=1}^{n} \delta_i^2 - \mathbb{E}[\delta_i^2 \mid \mathcal{D}^{\text{train}}] = o_p(1), \tag{95}$$

hence

$$\frac{1}{n} \sum_{i=1}^{n} \delta_i^2 = o_p(1). \tag{96}$$

Moreover, by Cauchy–Schwarz inequality, we have

$$\left| \frac{1}{n} \sum_{i=1}^{n} \psi_i \delta_i \right| \leq \left(\frac{1}{n} \sum_{i=1}^{n} \psi_i^2 \right)^{1/2} \left(\frac{1}{n} \sum_{i=1}^{n} \delta_i^2 \right)^{1/2} = o_p(1), \tag{97}$$

since $n^{-1} \sum_i \psi_i^2 \xrightarrow{p} \mathbb{E}[\psi_i^2]$ and Equation 96 holds. Therefore,

$$\frac{1}{n} \sum_{i=1}^{n} \tilde{\psi}_i^2 = \frac{1}{n} \sum_{i=1}^{n} \psi_i^2 + \frac{2}{n} \sum_{i=1}^{n} \psi_i \delta_i + \frac{1}{n} \sum_{i=1}^{n} \delta_i^2 \tag{98}$$

$$= \frac{1}{n} \sum_{i=1}^{n} \psi_i^2 + o_p(1) \xrightarrow{p} \mathbb{E}[\psi_i^2]. \tag{99}$$

Also, $\hat{\theta} \xrightarrow{p} \theta$, so

$$\frac{1}{n} \sum_{i=1}^{n} (\tilde{\psi}_i - \hat{\theta})^2 = \frac{1}{n} \sum_{i=1}^{n} \tilde{\psi}_i^2 - \hat{\theta}^2 \xrightarrow{p} \mathbb{E}[\psi_i^2] - \theta^2 = \mathbb{V}[\psi_i], \tag{100}$$

which proves Equation 94. Consequently,

$$\widehat{\mathbb{V}}(\hat{\theta}) = \frac{1}{n^2} \sum_{i=1}^{n} (\tilde{\psi}_i - \hat{\theta})^2 = \frac{1}{n} \left(\frac{1}{n} \sum_{i=1}^{n} (\tilde{\psi}_i - \hat{\theta})^2 \right) \tag{101}$$

satisfies

$$n\, \widehat{\mathbb{V}}(\hat{\theta}) \xrightarrow{p} \mathbb{V}[\psi_i]. \tag{102}$$

Therefore, $\widehat{\mathbb{V}}(\hat{\theta})$ is a consistent estimator of the asymptotic variance $\mathbb{V}[\psi_i]/n$ of $\hat{\theta}$. □

B.3 AUTOMATIC DML FOR OPERATOR-VALUED NUISANCES

Proposition B.6 (Decomposition of the debiasing weight)**.** *Suppose $\mathcal{U} = L^2(\Omega, \mu)$ and let $w_g(u) \in L^2(\Omega, \mu)$ denote the Riesz representer of $\mathrm{D}g_u$ with respect to μ. Then the unique solution $\beta_0(A)(x)$ satisfying Equation 11 admits ($\mathbb{P}(\cdot \mid A)$-a.e.) the decomposition*

$$\beta_0(A)(x) = \xi_0(A)(x)\, w_g(S_0(A))(x). \tag{103}$$

Proof. By definition of the Riesz representer, we have

$$\mathrm{D}g_{S_0(A)}(h) = \int_\Omega w_g(S_0(A))(x)\, h(x)\ \mathrm{d}\mu(x). \tag{104}$$

On the other hand,

$$\mathbb{E}[\beta_0(A)(X)\, h(X) \mid A] = \int_\Omega \beta_0(A)(x)\, h(x)\ \mathrm{d}\mathbb{P}(x \mid A). \tag{105}$$

Equating the two expressions for all $h \in \mathcal{H}_A$ implies

$$\beta_0(A)(x)\ \mathrm{d}\mathbb{P}(x \mid A) = w_g(S_0(A))(x)\ \mathrm{d}\mu(x), \tag{106}$$

which yields the decomposition in Equation 103 by Radon–Nikodym differentiation. □

Theorem B.7 (Primal characterization of the Riesz representer)**.** *The unique minimizer of Equation 22 is the Riesz representer β_0 satisfying Equation 20.*

Proof. We prove that the unique minimizer of the population risk

$$\mathcal{R}(\beta) \coloneqq \mathbb{E}\left[\mathbb{E}[\beta(A)(X)^2 \mid A] - 2\ \mathrm{D}g_{S_0(A)}\big(\beta(A)(\cdot)\big)\right] \tag{107}$$

is the Riesz representer β_0 satisfying

$$\mathbb{E}[\beta_0(A)(X)h(X) \mid A] = \mathrm{D}g_{S_0(A)}(h), \qquad \forall h \in \mathcal{H}_A. \tag{108}$$

Step 1: Conditional Hilbert space structure. For each $A = a$, define the Hilbert space

$$\mathcal{H}_a \coloneqq L^2(\mathbb{P}(\cdot \mid A = a)) \tag{109}$$

with inner product

$$\langle f, h\rangle_a \coloneqq \mathbb{E}[f(X)h(X) \mid A = a]. \tag{110}$$

By Assumption 3.1, for every $h \in \mathcal{H}_a$, we have

$$\|h\|_{\mathcal{U}}^2 = \int_\Omega h(x)^2\ \mathrm{d}\mu(x) = \int_\Omega h(x)^2\, \xi_0(a)(x)\ \mathrm{d}\mathbb{P}(x \mid a) \le C\, \|h\|_{\mathcal{H}_a}^2. \tag{111}$$

Hence, $\mathcal{H}_a$ embeds continuously into $\mathcal{U} = L^2(\Omega, \mu)$. Therefore, the Fréchet derivative

$$\mathcal{L}_a(h) \coloneqq \mathrm{D}g_{S_0(a)}(h) \tag{112}$$

defines a continuous linear functional on $\mathcal{H}_a$ by restriction.

Step 2: Existence of the Riesz representer. By the Riesz representation theorem, for each a, there exists a unique $\beta_0(a)(\cdot) \in \mathcal{H}_a$ such that

$$\mathcal{L}_a(h) = \langle \beta_0(a)(\cdot), h\rangle_a = \mathbb{E}[\beta_0(A)(X)h(X) \mid A = a], \qquad \forall h \in \mathcal{H}_a. \tag{113}$$

This establishes the conditional Riesz identity on $\mathcal{H}_a$.

Step 3: Variational characterization. We now use the standard variational characterization of the Riesz representer in a Hilbert space (Chernozhukov et al., 2021). For any Hilbert space $(\mathcal{H}, \langle\cdot,\cdot\rangle)$ and continuous linear functional $L(h) = \langle r, h\rangle$, the representer r satisfies

$$r = \arg\min_{b \in \mathcal{H}} \left\{\|b\|_{\mathcal{H}}^2 - 2L(b)\right\}. \tag{114}$$

Indeed, for any $b \in \mathcal{H}$, we obtain

$$\|b\|_{\mathcal{H}}^2 - 2L(b) = \|b\|_{\mathcal{H}}^2 - 2\langle r, b\rangle = \|b - r\|_{\mathcal{H}}^2 - \|r\|_{\mathcal{H}}^2, \tag{115}$$

which is uniquely minimized at $b = r$. This variational formulation is the basis of automatic Riesz regression in debiased machine learning (Chernozhukov et al., 2021; 2022a). Our construction extends this idea to a conditional, operator-valued setting.

Step 4: Conditional minimization. Applying Equation 114 with

$$\mathcal{H} = \mathcal{H}_a, \quad L = \mathcal{L}_a, \quad r = \beta_0(a)(\cdot), \tag{116}$$

we obtain, for each fixed a,

$$\beta_0(a)(\cdot) = \arg\min_{\beta(a)(\cdot) \in \mathcal{H}_a} \left\{\mathbb{E}[\beta(A)(X)^2 \mid A = a] - 2\ \mathrm{D}g_{S_0(a)}\big(\beta(a)(\cdot)\big)\right\}. \tag{117}$$

Step 5: Averaging over A. Finally, observe that the population risk Equation 107 can be written as

$$\mathcal{R}(\beta) = \mathbb{E}\Big[\mathcal{R}_A\big(\beta(A)(\cdot)\big)\Big], \tag{118}$$

where

$$\mathcal{R}_A\big(\beta(A)(\cdot)\big) \coloneqq \mathbb{E}[\beta(A)(X)^2 \mid A] - 2\ \mathrm{D}g_{S_0(A)}\big(\beta(A)(\cdot)\big). \tag{119}$$

Since the objective separates across values of A, minimizing $\mathcal{R}(\beta)$ is equivalent to minimizing $\mathcal{R}_A(\beta(A)(\cdot))$ for almost every A. By Equation 117, the unique minimizer is therefore $\beta_0(A)(\cdot)$.

This means that β_0 is the unique minimizer of Equation 107, which completes the proof.

□

C REAL-DATA EXPERIMENT ON ETTH1

We additionally evaluate DOPE on the ETTh1 electricity-transformer dataset (Zhou et al., 2021). This experiment complements the controlled simulations in the main paper with real input and output trajectories. Specifically, all covariate and oil temperature (OT) measurements are taken from the real dataset; only the sparse observation pattern of the future OT trajectory is imposed synthetically. This allows us to retain a complete future trajectory for evaluation while reproducing the partial observation setting considered throughout the paper.

Data and prediction task. We sort ETTh1 chronologically and use all seven numerical channels. Each sample consists of an input trajectory A_i containing the previous 96 hourly observations of all channels and an output trajectory $S_0(A_i)$ containing the subsequent 24 hourly OT measurements. We use a chronological split with 512 training, 64 validation, and 128 test trajectories.The plug-in estimator and DOPE use the *same frozen fitted solution operator* within each of the 50 seed repetitions.

Functionals. We consider two functionals of the future OT trajectory. First, we use the time-normalized 24-hour area under the curve. Thus, in this experiment AUC is numerically equal to the average future OT rather than the unnormalized degree-hour integral. Second, we consider a smooth approximation of the daily maximum via soft-cmax.

Sparse observation design. To emulate irregularly observed future trajectories, exactly $K = 8$ of the 24 future OT measurements are retained for each sample. Locations are sampled without replacement using the categorical draw weights

$$p_j(\rho) = \frac{1-\rho}{24} + \rho q_j, \tag{120}$$

where

$$q_j = \begin{cases} 0.02/24 + 0.98/8, & j = 1, \dots, 8, \\ 0.02/24, & j = 9, \dots, 24, \end{cases} \tag{121}$$

for $\rho \in \{0, 0.25, 0.5, 0.75, 1\}$. At $\rho = 0$, all 24 future locations have equal draw weight $1/24$. Increasing ρ progressively concentrates observations in the first eight hours of the forecast horizon.

	$\rho = 0$		$\rho = 0.25$		$\rho = 0.5$		$\rho = 0.75$		$\rho = 1$	
	AUC	Soft Cmax	AUC	Soft Cmax	AUC	Soft Cmax	AUC	Soft Cmax	AUC	Soft Cmax
Plug-in	1.226 ± 0.043	1.171 ± 0.041	1.210 ± 0.043	1.181 ± 0.043	1.224 ± 0.044	1.215 ± 0.044	1.310 ± 0.057	1.340 ± 0.062	1.976 ± 0.094	2.212 ± 0.117
DOPE	0.112 ± 0.010	0.239 ± 0.014	0.128 ± 0.013	0.264 ± 0.020	0.167 ± 0.017	0.305 ± 0.022	0.178 ± 0.016	0.316 ± 0.020	0.638 ± 0.094	0.874 ± 0.149
Rel. improvement	90.8%	79.6%	89.5%	77.7%	86.3%	74.9%	86.4%	76.4%	67.7%	60.5%

Table 4: **ETTh1 real-data results.** RMSE $\pm$ SE over 50 repetitions for normalized AUC (mean future OT) and Soft Cmax under increasingly concentrated sparse-observation designs. The underlying transformer trajectories are real; only the future observation locations are synthetically subsampled. $\Rightarrow$ *DOPE improves over the plug-in estimator for both functionals and at every sampling-irregularity level.*

Results. Table 4 show that, across both functionals, DOPE achieves lower RMSE than the plug-in estimator for every value of ρ. For normalized AUC, the improvement ranges from 90.8% under uniform sampling to 67.7% under the most strongly concentrated early-block design. For soft cmax, the corresponding improvement ranges from 79.6% to 60.5%.

As ρ increases, observations become progressively concentrated in the beginning of the 24-hour forecast window and provide decreasingly direct information about later hours. Accordingly, estimation becomes more difficult for both methods, particularly at $\rho = 1$. Nevertheless, DOPE retains a substantial advantage throughout.

These results show that DOPE is directly applicable to real-world data. Hence, the gains from the proposed debiasing procedure extend beyond the simulated pharmacokinetic and PDE settings: when applied to real transformer trajectories, DOPE consistently reduces functional estimation error for both a linear trajectory summary and a nonlinear peak functional.

D ADDITIONAL RESULTS

D.1 EXPERIMENTS WITH DEEPONET BACKBONE

We repeat our main experimental results from Section 5 with DeepONet (Lu et al., 2021) as the neural backbone of both DOPE and the plug-in estimator. The new instantiation confirms our previous results: Table 5 DOPE clearly improves upon the plug-in estimator for both functionals and all sampling levels. At the same time, this also shows that DOPE is broadly applicable across different neural operator architectures.

	$\rho = 0$		$\rho = 0.125$		$\rho = 0.25$		$\rho = 0.375$		$\rho = 0.5$		$\rho = 0.625$		$\rho = 0.75$		$\rho = 0.875$		$\rho = 1$	
	AUC	TAT	AUC	TAT	AUC	TAT	AUC	TAT	AUC	TAT	AUC	TAT	AUC	TAT	AUC	TAT	AUC	TAT
DOPE: structured w_q (ours)	0.80 ± 0.08	6.39 ± 0.10	0.69 ± 0.07	6.13 ± 0.11	0.89 ± 0.08	6.51 ± 0.11	1.08 ± 0.10	6.33 ± 0.12	0.86 ± 0.08	6.41 ± 0.12	0.76 ± 0.07	6.37 ± 0.12	0.92 ± 0.10	6.42 ± 0.13	0.84 ± 0.08	6.04 ± 0.13	0.87 ± 0.09	6.45 ± 0.13
DOPE: oracle β_0 (ours)	0.72 ± 0.06	0.95 ± 0.07	0.73 ± 0.09	0.84 ± 0.07	0.80 ± 0.07	0.92 ± 0.08	1.02 ± 0.11	1.02 ± 0.07	0.82 ± 0.07	0.97 ± 0.10	0.75 ± 0.07	0.96 ± 0.07	0.90 ± 0.09	1.06 ± 0.10	0.88 ± 0.09	1.06 ± 0.09	0.85 ± 0.09	0.93 ± 0.09
Plug-in	1.78 ± 0.21	6.78 ± 0.18	1.70 ± 0.16	6.93 ± 0.17	1.90 ± 0.15	6.79 ± 0.19	1.93 ± 0.19	6.74 ± 0.20	1.77 ± 0.15	7.05 ± 0.18	1.48 ± 0.13	7.07 ± 0.15	2.03 ± 0.18	7.23 ± 0.19	2.31 ± 0.22	6.87 ± 0.23	2.06 ± 0.18	6.79 ± 0.20
DOPE (**ours**)	0.73 ± 0.07	6.34 ± 0.11	0.73 ± 0.09	6.11 ± 0.11	0.83 ± 0.07	6.52 ± 0.11	1.05 ± 0.10	6.36 ± 0.13	0.84 ± 0.07	6.49 ± 0.11	0.76 ± 0.07	6.45 ± 0.12	0.91 ± 0.10	6.56 ± 0.12	0.86 ± 0.09	6.21 ± 0.13	0.88 ± 0.09	6.61 ± 0.13
Rel. improvement	58.9%	6.5%	57.2%	11.8%	56.5%	4.0%	45.6%	5.7%	52.6%	7.9%	49.1%	8.7%	55.3%	9.2%	62.8%	9.6%	57.4%	2.6%

Table 5: **Comparison: plug-in estimator vs. DOPE.** RMSE $\pm$ SE ($\times 100$) across different functionals (area under the concentration curve [AUC] and time-above-threshold [TAT]) for different overlap levels ρ. $\Rightarrow$ *DOPE outperforms the plug-in estimator.*

D.2 AUC FUNCTIONAL FOR PPI-STYLE EXPERIMENTS

We repeat our PPI-style experiments from Section 5 on the AUC functional. Our results in Table 6 are consistent with the smooth continuous maximum functional. Both DOPE and the plug-in estimator benefit from abundant unlabeled trajectories, but DOPE clearly has stronger performance gains.

	$n_2 = 0$	$n_2 = 64$	$n_2 = 128$	$n_2 = 256$	$n_2 = 512$	$n_2 = 1024$	$n_2 = 2048$	$n_2 = 4096$
Plug-in	1.06 ± 0.09	1.05 ± 0.09	0.99 ± 0.08	0.92 ± 0.08	0.90 ± 0.07	0.89 ± 0.08	0.84 ± 0.07	0.77 ± 0.06
Rel. improvement	0.0%	1.1%	6.5%	12.8%	14.9%	15.6%	20.6%	26.9%
DOPE	0.72 ± 0.06	0.70 ± 0.06	0.59 ± 0.05	0.55 ± 0.05	0.48 ± 0.05	0.43 ± 0.04	0.36 ± 0.04	0.30 ± 0.03
Rel. improvement	0.0%	2.4%	17.6%	23.3%	33.6%	40.6%	50.1%	58.6%

Table 6: AUC RMSE $\pm$ SE ($\times 100$) across unlabeled sample sizes n_2.

D.3 CLT-BASED CONFIDENCE INTERVALS

We report the empirical coverage for CLT-based confidence intervals in Table 7 on the AUC functional with the pharmacokinetics data. Using the CLT is grounded in Proposition B.5. We find that DOPE is fairly reliable, while the empirical coverage from simple plug-in estimation is not.

	$\rho = 0$		$\rho = 0.125$		$\rho = 0.25$		$\rho = 0.375$		$\rho = 0.5$		$\rho = 0.625$		$\rho = 0.75$		$\rho = 0.875$		$\rho = 1$	
	Cov.	Length	Cov.	Length	Cov.	Length	Cov.	Length	Cov.	Length	Cov.	Length	Cov.	Length	Cov.	Length	Cov.	Length
Plug-in	0.68	0.02554	0.88	0.02555	0.78	0.02517	0.76	0.02530	0.76	0.02500	0.80	0.02543	0.76	0.02510	0.76	0.02569	0.78	0.02496
DOPE (ours)	0.94	0.02747	0.94	0.02792	0.94	0.02736	0.92	0.02738	0.90	0.02729	0.96	0.02768	0.94	0.02752	0.96	0.02804	0.92	0.02749

Table 7: Empirical coverage and mean confidence interval length for 95% CLT-based confidence intervals across overlap levels ρ over 50 repeats.

E PSEUDO-CODE FOR AUTOMATIC DML WITH NEURAL OPERATORS

Algorithm 1 Cross-fitted debiased neural operator

Input: Observations $\mathcal{D} = \{O_i\}_{i=1}^n$, where $O_i = (A_i, K_i, \{(X_{ik}, Y_{ik})\}_{k=1}^{K_i})$; number of folds J; common discretization $\{x_\ell\}_{\ell=1}^m \subset \Omega$; discretized functional $g_m : \mathbb{R}^m \to \mathbb{R}$; regularization parameter $\lambda \geq 0$; neural operator model classes $\mathcal{S}_0$ for S and $\mathcal{B}$ for β_0

Output: One-step estimate $\hat{\theta}$

Split $\{1, \ldots, n\}$ into disjoint folds $I_1, \ldots, I_J$

for $j = 1, \ldots, J$ **do**

Define training and evaluation indices

$$I_j^{\text{train}} \leftarrow \{1, \ldots, n\} \setminus I_j, \qquad I_j^{\text{eval}} \leftarrow I_j \tag{122}$$

Train neural operator for the solution map:

$$\hat{S}^{(-j)} \leftarrow \arg\min_{S_\eta \in \mathcal{S}} \frac{1}{|I_j^{\text{train}}|} \sum_{i \in I_j^{\text{train}}} \frac{1}{K_i} \sum_{k=1}^{K_i} \big(Y_{ik} - S(A_i)(X_{ik})\big)^2 \tag{123}$$

Instantiate the debiasing-weight model as a neural operator: let $\beta_\eta \in \mathcal{B}$ be a neural operator mapping

$$A \mapsto \beta_\eta(A, \cdot) \tag{124}$$

Optionally, if w_g is known in closed form, use the structured parameterization

$$\beta_\eta(A)(x) = \xi_\eta(A)(x)\, w_g(\hat{S}^{(-j)}(A))(x) \tag{125}$$

with ξ_η parameterized by a neural operator.

Train neural operator for the debiasing weight via Riesz regression:

$$\hat{\beta}^{(-j)} \leftarrow \arg\min_{\beta_\eta \in \mathcal{B}} \frac{1}{|I_j^{\text{train}}|} \sum_{i \in I_j^{\text{train}}} \left[\frac{1}{K_i} \sum_{k=1}^{K_i} \beta_\eta(A_i)(X_{ik})^2 - 2\,\mathrm{JVP}(g_m, u_i; b_i) \right] + \lambda \|\eta\|^2 \tag{126}$$

where

$$u_i \leftarrow \big(\hat{S}^{(-j)}(A_i)(x_\ell)\big)_{\ell=1}^m \in \mathbb{R}^m, \qquad b_i \leftarrow \big(\beta_\eta(A_i)(x_\ell)\big)_{\ell=1}^m \in \mathbb{R}^m, \tag{127}$$

and

$$\mathrm{JVP}(g_m, u_i; b_i) \leftarrow \frac{\mathrm{d}}{\mathrm{d}t} g_m(u_i + t b_i)\Big|_{t=0}. \tag{128}$$

Evaluate pseudo-outcomes on the held-out fold:

for each $i \in I_j^{\text{eval}}$ **do**

Compute

$$\tilde{\psi}_i \leftarrow g_m\Big(\big(\hat{S}^{(-j)}(A_i)(x_\ell)\big)_{\ell=1}^m\Big) + \frac{1}{K_i} \sum_{k=1}^{K_i} \hat{\beta}^{(-j)}(A_i)(X_{ik}) \big(Y_{ik} - \hat{S}^{(-j)}(A_i)(X_{ik})\big). \tag{129}$$

end for

end for

Aggregate cross-fitted estimate:

$$\hat{\theta} \leftarrow \frac{1}{n} \sum_{i=1}^n \tilde{\psi}_i \tag{130}$$

return One-step estimate $\hat{\theta}$

F EMPIRICAL OBJECTIVE OF DOPE VIA AUTOMATIC DIFFERENTIATION

Empirical objective and implementation of DOPE

We directly parameterize the correction weight $\hat{\beta}(\cdot)(\cdot)$ using a neural operator, and minimize the empirical analogue of Equation 22. Specifically, let $\hat{S}$, $\hat{\beta}$ denote a cross-fitted estimators of S_0, β_0, respectively. For each observation A_i, we evaluate both $\hat{S}(A_i)(\cdot)$ and $\hat{\beta}(A_i)(\cdot)$ on a common discretization $\{x_\ell\}_{\ell=1}^m \subset \Omega$, which yields

$$u_i \coloneqq \big(\hat{S}(A_i)(x_\ell)\big)_{\ell=1}^m \in \mathbb{R}^m, \qquad b_i \coloneqq \big(\hat{\beta}(A_i)(x_\ell)\big)_{\ell=1}^m \in \mathbb{R}^m. \tag{131}$$

Let $g_m : \mathbb{R}^m \to \mathbb{R}$ denote the discretized implementation of the functional g.[5]The empirical risk is then given by

$$\min_{\hat{\beta}} \frac{1}{n}\sum_{i=1}^{n}\left[\frac{1}{K_i}\sum_{k=1}^{K_i}\hat{\beta}(A_i)(X_{ik})^2 - 2\,\mathrm{JVP}(g_m, u_i; b_i)\right] + \lambda\|\hat{\beta}\|^2, \tag{132}$$

where the directional derivative is computed as a Jacobian–vector product

$$\mathrm{JVP}(g_m, u_i; b_i) = \frac{\mathrm{d}}{\mathrm{d}t} g_m(u_i + t\,b_i)\Big|_{t=0} = J_{g_m}(u_i)\,b_i. \tag{133}$$

This quantity can be evaluated efficiently via automatic differentiation (e.g., in forward mode) and thus *without* explicitly forming the Jacobian matrix.

Importantly, even though the objective of DOPE in Equation 22 is formulated in function space, our empirical implementation in DOPE reduces to differentiating the scalar computational graph $t \mapsto g_m(u_i + tb_i)$ at $t = 0$. In particular, DOPE requires only that both $\hat{S}(A)(\cdot)$ and $\hat{\beta}(A)(\cdot)$ can be evaluated on a common discretization of Ω, which is naturally satisfied by neural operator architectures.

[5]Of note, while our theory is formulated in function space for the continuous functional g, the implemented estimator targets a numerical approximation to the population quantity. The resulting discretization and quadrature error is separate from the statistical error analyzed here.

G IMPLEMENTATION DETAILS

• **Neural operator implementation.** To make the comparison in Section 5 and Supplement D fair, DOPE and the plug-in baseline use the *exact same* neural backbone. Specifically, the plug-in estimator only consists of the solution operator $\hat{S}$, which is shared for both methods (see Table 8).

Setting	Operator(s)	Backbone	Architecture specification
Pharmacokinetics (main)	$\hat{S}, \hat{\beta}$	FNO (Li et al., 2021)	4 input channels, 32 hidden channels, 1 output channel, 3 Fourier layers, 12 retained modes
2D Darcy-flow (main)	$\hat{S}, \hat{\beta}$	FNO (Li et al., 2021)	3 input channels, 24 hidden channels, 1 output channel, 3 Fourier layers, 8×8 retained modes
Pharmacokinetics (ablation)	$\hat{S}, \hat{\beta}$	DeepONet (Lu et al., 2021)	4 input channels, 32 branch hidden channels, 32 trunk hidden channels, latent dimension 32, 1 output channel

Table 8: Neural operator architectures used in the main experiments and ablation study. For fairness, DOPE and the plug-in baseline share the same solution-operator backbone $\hat{S}$ within each setting.

Pharmacokinetics dataset: Both nuisance components, namely the solution operator $\hat{S}$ and the debiasing operator $\hat{\beta}$, are implemented using the same Fourier neural operator (FNO) (Li et al., 2021) architecture. The shared model configuration uses the FNO family with 4 input channels, 32 hidden channels, 1 output channel, 3 Fourier layers, and 12 retained modes. Both modules are trained with Adam optimization (Kingma & Ba, 2015) settings given by learning rate 10^{-3}, weight decay 10^{-5}, batch size 8, and a maximum of 20 epochs (see Table 9).

2D Darcy-flow dataset: Here, $\hat{S}$ and $\hat{\beta}$ are also implemented with the same Fourier neural architecture. Specifically, they use the FNO family with 3 input channels, 24 hidden channels, 1 output channel, 3 Fourier layers, and 8 retained modes for both x- and y-dimension, respectively. Both are learned with Adam optimization (Kingma & Ba, 2015) and learning rate 10^{-3}, weight decay 10^{-5}, batch size 8, and a maximum of 20 epochs (see Table 9).

Ablation instantiation: In Supplement D, we report a variant of our main results on the pharmacokinetics dataset, where we replace the Fourier neural operator with a DeepONet (Lu et al., 2021)). For both the solution operator $\hat{S}$ of the plug-in estimator and DOPE, as well as the debiasing operator $\hat{\beta}$ of DOPE, the DeepONets have 4 input channels, 32 branch hidden channels, 32 trunc hidden channels, a latent dimension of 32, and 1 output channel. They are trained via Adam (Kingma & Ba, 2015) with a learning rate of 10^{-3}, a weight decay of 10^{-5}, batch size 8, and a maximum of 20 epochs (see Table 9).

Component	Setting	Specification
Optimizer	all models	Adam-style optimization with learning rate 10^{-3} and weight decay 10^{-5}
Batch size / epochs	all models	batch size 8, maximum of 20 epochs
Solution operator training	$\hat{S}$	trained to predict the latent trajectory from A_i using mean squared error evaluated only at the sampled observation locations
Debiasing operator training	$\hat{\beta}$ for DOPE	trained separately via the implemented Riesz-style objective with the same optimizer hyperparameters as $\hat{S}$
Riesz penalty	$\hat{\beta}$ for DOPE	$\lambda_{\mathrm{Riesz}} = 0.001$ (PK) / 0.01 (Darcy)
Cross-fitting	final experiments	2-fold cross-fitting on the test split
Repeated runs	final experiments	results aggregated over 50 repetitions

Table 9: Training and evaluation details used throughout the experiments.

• **Solution operator training.** For both DOPE and the plug-in estimator, the neural operator $\hat{S}$ is trained to predict the full latent concentration trajectory from the functional input A_i. Its loss is the mean squared prediction error evaluated only at the subject-specific sampled observation times.

• **Debiasing operator training.** For DOPE, the correction operator $\hat{\beta}$ is parameterized by the same FNO architecture, and trained separately through the implemented Riesz-style objective used by the one-step estimator. The corresponding debiasing module uses the same optimizer hyperparameters as the solution module and includes the penalty parameter $\lambda_{\mathrm{Riesz}} = 0.1$.

• **Cross-fitting and evaluation.** The final experiments use 2-fold cross-fitting on the test split. The reported results aggregate repeated-run summaries over 50 repetitions.

• **Runtime.** For each method, training took approximately 30 minutes over 50 repeats with an AMD Ryzen 7 Pro CPU and 32GB of RAM. The runtime was comparable for DOPE and the plug-in estimator.

H ADDITIONAL DETAILS ON THE DATA

H.1 PHARMACOKINETIC MODEL

We consider a pharmacokinetics benchmark in which each subject is characterized by a function-valued input A_i consisting of a subject-specific dosing-rate profile together with subject-level clearance and volume parameters. The resulting latent concentration trajectory $u(A_i, \cdot)$ follows a standard one-compartment pharmacokinetic model (Kwon, 2002; Upton et al., 2016), where clearance governs elimination, volume determines how the administered drug is translated into concentration, and the time-varying dosing profile allows for heterogeneous infusion- or pulse-like regimens across subjects.

The corresponding scientific targets are not the full concentration trajectories themselves, but rather low-dimensional summaries such as exposure, peak burden, or time above a clinically relevant threshold. We observe each trajectory only at sparse time points and vary the degree of input-dependent sampling through an irregularity parameter ρ, where $\rho = 0$ corresponds to approximately uniform monitoring and larger values of ρ increasingly concentrate observations near the subject-specific peak region.

We now introduce the exact data-generating process for our pharmacokinetics benchmark study. We summarize all parameters in Table 10.

Table 10: Overview of the pharmacokinetics benchmark configuration used in the main study.

Component	Specification
Time horizon	$T = 24$
Grid size	$\Delta = 128$
Reference measure	normalized trapezoid rule on $[0, T]$
Input channels	$[r_i(t), \log CL_i, \log V_i]$
PK parameter means	$(0.0, 1.0)$ for $(\log CL_i, \log V_i)$
PK parameter standard deviations	$(0.25, 0.20)$
PK parameter correlation	0.4
Dosing family	`one_or_two_pulse`
Number of pulses	$M_i \in \{1, 2\}$ uniformly
Pulse amplitude range	$[0.5, 2.0]$
Pulse duration range	$[1.0, 4.0]$
Observation design	`peak_window_mixture`
Observations per subject	$K = 24$
Noise standard deviation	$\sigma_\varepsilon = 0.002$
Peak bandwidth	$h = 5.0$
Irregularity grid	$\rho \in \{0.00, 0.125, \ldots, 1.00\}$
Dataset split sizes	$(256, 64, 256)$ for train/val/test
Repeated runs	50
Cross-fitting	2 folds

• **Governing model:** We consider a one-dimensional pharmacokinetics benchmark on the time horizon $[0, T]$ with $T = 24$. The latent concentration trajectory u_i for subject i solves the one-compartment dynamics

$$\frac{\mathrm{d}u_i}{\mathrm{d}t}(t) = -\frac{CL_i}{V_i}u_i(t) + \frac{1}{V_i}r_i(t), \qquad u_i(0) = 0, \tag{134}$$

where r_i is a subject-specific dosing-rate profile and (CL_i, V_i) are clearance and volume parameters.

At the operator-learning level, each subject is represented through the function-valued input

$$A_i(t_\delta) = \big[r_i(t_\delta), \log CL_i, \log V_i\big] \in \mathbb{R}^3, \qquad \delta = 1, \ldots, \Delta, \tag{135}$$

where the last two channels are constant over the grid. We use a uniform grid of size $\Delta = 128$ with points

$$t_\delta = \frac{\delta - 1}{\Delta - 1}T, \qquad \delta = 1, \ldots, \Delta. \tag{136}$$

The subject-specific PK parameters are sampled as

$$\begin{pmatrix} CL_i^{\log} \\ V_i^{\log} \end{pmatrix} \sim \mathcal{N}\left(\begin{pmatrix} 0.0 \\ 1.0 \end{pmatrix}, \Sigma \right), \qquad \Sigma = \begin{pmatrix} 0.25^2 & 0.4 \cdot 0.25 \cdot 0.20 \\ 0.4 \cdot 0.25 \cdot 0.20 & 0.20^2 \end{pmatrix}, \tag{137}$$

and we set

$$CL_i = \exp(CL_i^{\log}), \qquad V_i = \exp(V_i^{\log}). \tag{138}$$

The dosing profile follows the `one_or_two_pulse` family. Specifically,

$$M_i \sim \mathrm{Unif}\{1, 2\}. \tag{139}$$

For each pulse $m = 1, \ldots, M_i$, we draw independently

$$D_{im} \sim \mathrm{Unif}[1.0, 4.0], \qquad S_{im} \sim \mathrm{Unif}[0, T - D_{im}], \qquad c_{im} \sim \mathrm{Unif}[0.5, 2.0]. \tag{140}$$

The dosing-rate profile is then

$$r_i(t_\delta) = \sum_{m=1}^{M_i} c_{im}\, \mathbf{1}\{S_{im} \le t_\delta \le \min\{T, S_{im} + D_{im}\}\}\,. \tag{141}$$

• **Implementation level:** Trajectories are generated on the grid using the recursion

$$u_{i,1} = 0, \tag{142}$$

and, for $\delta = 2, \ldots, \Delta$,

$$u_{i,\delta} = e^{-\kappa_i \Delta t} u_{i,\delta-1} + \frac{r_i(t_{\delta-1})}{V_i} \frac{1 - e^{-\kappa_i \Delta t}}{\kappa_i}, \qquad \kappa_i = \frac{CL_i}{V_i}, \tag{143}$$

with the small-κ_i fallback

$$u_{i,\delta} = u_{i,\delta-1} + \Delta t \frac{r_i(t_{\delta-1})}{V_i}. \tag{144}$$

Quadrature uses normalized trapezoid weights

$$w_1 = w_\Delta = \frac{1}{2(\Delta - 1)}, \qquad w_\delta = \frac{1}{\Delta - 1}, \quad \delta = 2, \ldots, \Delta - 1, \tag{145}$$

so that $\sum_{\delta=1}^{\Delta} w_\delta = 1$. The corresponding discrete inner product is

$$\langle f, h \rangle_w = \sum_{\delta=1}^{\Delta} w_\delta f_\delta h_\delta. \tag{146}$$

Sampling mechanism and irregularity parameter. We observe each latent trajectory only at *sparse* time points. In all reported pharmacokinetics experiments, each subject contributes $K = 24$ observations. The latent trajectory is represented on an equally spaced grid $\{t_\delta\}_{\delta=1}^{\Delta}$ over $[0, 24]$, with $\Delta = 128$.

We define the fixed early-time window mass function

$$q_\delta^{\mathrm{early}} = \frac{\mathbf{1}\{|t_\delta - c| \le h\}}{\sum_{\ell=1}^{\Delta} \mathbf{1}\{|t_\ell - c| \le h\}}, \qquad c = 1.0, \qquad h = 1.0. \tag{147}$$

Thus, q^{early} is uniform over the grid points in the interval $[0, 2]$ and assigns zero mass elsewhere. The observation design is the mixture

$$p_\delta(\rho) = \{1 - \gamma(\rho)\} \frac{1}{\Delta} + \gamma(\rho) q_\delta^{\mathrm{early}}, \qquad \gamma(\rho) = 0.45\sqrt{\rho}. \tag{148}$$

Hence, $\rho = 0$ corresponds to uniform monitoring over the grid, while larger values of ρ increasingly concentrate observations in the fixed early-time window. At $\rho = 1$, the early-window component receives mixture weight 0.45. In the main study, we use

$$\rho \in \{0.00, 0.125, 0.25, 0.375, 0.50, 0.625, 0.75, 0.875, 1.00\}. \tag{149}$$

Given $p_\delta(\rho)$, we draw

$$J_{ik} \overset{\text{iid}}{\sim} \text{Categorical}\{p_1(\rho), \dots, p_\Delta(\rho)\}, \qquad k = 1, \dots, K, \tag{150}$$

with replacement, sort the resulting indices, and observe

$$Y_{ik} = u_i(t_{J_{ik}}) + \varepsilon_{ik}, \qquad \varepsilon_{ik} \sim \mathcal{N}(0, \sigma_\varepsilon^2), \qquad \sigma_\varepsilon = 0.002. \tag{151}$$

For **oracle calculations**, the inverse-design weight on the grid is

$$\xi_i(t_\delta) = \frac{w_\delta}{p_\delta(\rho)}, \tag{152}$$

where w_δ denotes the normalized trapezoidal quadrature weight on the time grid.

Functional	Definition	Implementation	Hyperparameters
`auc`	$g(u) = (1/T)\int_0^T u(t)\,\mathrm{d}t$	$g(u) = \sum_{\delta=1}^{\Delta} w_\delta u_\delta$	none
`tat`	$g(u) = (1/T)\int_0^T \sigma(\kappa(u(t) - c^\star))\,\mathrm{d}t$	$g(u) = \sum_{\delta=1}^{\Delta} w_\delta\, \sigma\big(\kappa(u_\delta - c^\star)\big)$	$\kappa = 8.0,\ c^\star = 0.5$
`soft_cmax`	$g(u) = (1/\lambda)\log\left[(1/T)\int_0^T \exp(\lambda u(t))\,\mathrm{d}t\right]$	$g(u) = \frac{1}{\lambda}\log\left(\sum_{\delta=1}^{\Delta} w_\delta e^{\lambda u_\delta}\right)$	$\lambda = 6.0$

Table 11: Functional-specific hyperparameters used in the reported pharmacokinetics experiments.

• **Functional estimation:** The population target is the mean functional

$$\theta = \mathbb{E}[g(S_0(A))], \tag{153}$$

where we use 2000 datapoints to compute the ground-truth average functional.

Table 11 lists the functional-specific hyperparameters. More details on the closed-form functional derivatives and their Riesz representers are given in Supplement I. All experiments are repeated 50 times.

H.2 DARCY-FLOW MODEL

We consider a two-dimensional Darcy-flow benchmark in which each sample is characterized by a spatially varying coefficient field a_i on the unit square, and the latent scientific state is the corresponding solution field $u(a_i, \cdot)$ of an elliptic PDE. The operator-learning task is therefore to map coefficient fields to solution fields, but the inferential target is not the full field itself: instead, we study low-dimensional functionals of the solution, namely smooth excess-above-threshold summaries of the spatial field. The Darcy-flow model is *standard for benchmarking neural operators* (Kovachki et al., 2023).

We observe each latent solution only at a sparse set of noisy spatial locations and allow the observation design to depend on the realized coefficient and solution field. The observation design is fully adaptive, and concentrates samples in regions with small coefficients and rapidly varying solutions.

We now introduce the exact data-generating process for our Darcy-flow benchmark study. We summarize all benchmark parameters in Table 12.

Table 12: Overview of the Darcy-flow benchmark configuration used in the main study.

Component	Specification
Spatial domain	$\Omega = (0,1)^2$
Grid size	$H = W = 17$
Reference measure	normalized 2D trapezoid rule on Ω
Coefficient family	`piecewise_constant_transformed_gaussian_field`
Coarse coefficient grid	5×5
Number of cosine basis modes	3×3
Log-coefficient scale	0.65
Lower coefficient bound	0.2
Observation design	`inverse_energy_mixture`
Observations per sample	$K = 24$
Noise standard deviation	$\sigma_\varepsilon = 0.01$
Adaptive design temperature	$\lambda = 7.0$
Adaptive design floor	$\varepsilon = 10^{-3}$
Irregularity parameter	$\rho = 1.0$
Dataset split sizes	$(256, 64, 64)$ for train/val/test
Truth-pool size	2000
Repeated runs	50
Cross-fitting	2 folds

• **Governing model:** We consider the elliptic Darcy problem on $\Omega = (0,1)^2$,

$$-\nabla \cdot \big(a_i(x)\nabla u_i(x)\big) = 1, \qquad x \in (0,1)^2, \tag{154}$$

with homogeneous Dirichlet boundary conditions

$$u_i(x) = 0, \qquad x \in \partial(0,1)^2. \tag{155}$$

Thus, the forcing term is constant and equal to 1, and all sample-specific heterogeneity enters through the random coefficient field a_i.

At the operator-learning level, each sample is represented by a coefficient field on a tensor-product grid with

$$x_p = \frac{p-1}{H-1}, \qquad p = 1, \dots, H, \qquad y_q = \frac{q-1}{W-1}, \qquad q = 1, \dots, W, \tag{156}$$

where $H = W = 17$, and grid locations

$$X_{pq} = (x_p, y_q) \in [0,1]^2. \tag{157}$$

The coefficient field is sampled from the family `piecewise_constant_transformed_gaussian_field`. First, define a coarse 5×5 grid by

$$\bar{x}_r = \frac{r-1}{5-1}, \qquad r = 1, \dots, 5, \qquad \bar{y}_s = \frac{s-1}{5-1}, \qquad s = 1, \dots, 5. \tag{158}$$

For each pair of modes $(m_x, m_y) \in \{1, 2, 3\}^2$, we use the cosine basis

$$\phi_{m_x,m_y}(\bar{x}_r, \bar{y}_s) = \sqrt{2}\cos(\pi m_x \bar{x}_r)\,\sqrt{2}\cos(\pi m_y \bar{y}_s). \tag{159}$$

Independent Gaussian coefficients $\xi_{m_x,m_y} \sim \mathcal{N}(0, 1)$ are drawn, and the latent coarse field is

$$Z_{rs} = \frac{0.65}{3} \sum_{m_x=1}^{3} \sum_{m_y=1}^{3} \xi_{m_x,m_y}\, \phi_{m_x,m_y}(\bar{x}_r, \bar{y}_s). \tag{160}$$

The coarse coefficient field is then obtained by exponentiation and lower clipping,

$$\bar{a}_{rs} = \exp(Z_{rs}), \qquad \bar{a}_{rs} \leftarrow \max\{\bar{a}_{rs}, 0.2\}. \tag{161}$$

Finally, the coarse field is mapped to the 17×17 grid by nearest-cell replication in both spatial directions, yielding a piecewise-constant coefficient field on the fine grid.

• **Implementation level:** The PDE is solved on the 17×17 grid by a finite-difference discretization of the divergence-form operator. Let

$$h_x = h_y = \frac{1}{17 - 1}. \tag{162}$$

On interior grid points, the code assembles the linear system corresponding to

$$-\partial_x(a\,\partial_x u) - \partial_y(a\,\partial_y u) = 1. \tag{163}$$

To preserve flux continuity across coefficient jumps, harmonic means are used at cell interfaces. For example, at the west interface,

$$a^{(h)}_{i-\frac{1}{2},j} = \frac{2a_{ij}a_{i-1,j}}{a_{ij} + a_{i-1,j}}, \tag{164}$$

with analogous formulas for east, north, and south interfaces.

The resulting interior system has dimensions

$$(17 - 2) \times (17 - 2) = 225, \tag{165}$$

is assembled explicitly as a dense matrix, and is solved using `numpy.linalg.solve`. Boundary values are then set to zero to obtain the full 17×17 solution field.

Quadrature uses a normalized two-dimensional trapezoid rule. If $w^{(x)}$ and $w^{(y)}$ denote the one-dimensional trapezoid weights,

$$w^{(x)}_1 = w^{(x)}_H = \frac{1}{2}, \qquad w^{(x)}_p = 1 \text{ for } p = 2, \ldots, H - 1, \tag{166}$$

and analogously in the y-direction, then the unnormalized two-dimensional weights are

$$\widetilde{w}_{pq} = w^{(x)}_p w^{(y)}_q. \tag{167}$$

These are normalized to sum to one, i.e.,

$$w_{pq} = \frac{\widetilde{w}_{pq}}{\sum_{r=1}^{H}\sum_{s=1}^{W}\widetilde{w}_{rs}}. \tag{168}$$

The corresponding discrete inner product is

$$\langle f, h\rangle_w = \sum_{p=1}^{H}\sum_{q=1}^{W} w_{pq} f_{pq} h_{pq}. \tag{169}$$

Sampling mechanism and irregularity parameter. We observe each latent solution field only at a sparse set of spatial locations. In all reported Darcy-flow experiments, each sample contributes

$$K = 24 \tag{170}$$

noisy observations. The solution is represented on a 17×17 spatial grid. Let a_{pq} denote the coefficient field and u_{pq} the corresponding latent solution at grid location x_{pq}.

The observation design is `inverse_energy_mixture`. First, we define the inverse-coefficient factor

$$a_{pq}^{-1} = \frac{1}{\max\{a_{pq}, 10^{-8}\}}. \tag{171}$$

We then compute the discrete solution-gradient magnitude

$$G_{pq} = \Big[\big(u_{p+1,q} - u_{p-1,q}\big)^2 + \big(u_{p,q+1} - u_{p,q-1}\big)^2\Big]^{1/2}, \tag{172}$$

where grid indices are interpreted periodically, as in the implementation. The inverse-energy saliency map is

$$S_{pq} = a_{pq}^{-1}\big\{0.5 + |u_{pq}| + G_{pq}\big\}. \tag{173}$$

After centering and normalizing this field,

$$\widetilde{S}_{pq} = \frac{S_{pq} - \overline{S}}{\max_{r,s} |S_{rs} - \overline{S}|}, \tag{174}$$

with $\widetilde{S}_{pq} = 0$ if the denominator is zero, we define the adaptive design distribution

$$q_{pq} = \frac{\exp\big\{7|\widetilde{S}_{pq}|\big\} + 10^{-3}}{\sum_{r,s}\Big[\exp\big\{7|\widetilde{S}_{rs}|\big\} + 10^{-3}\Big]}. \tag{175}$$

Let $G_x = G_y = 17$ denote the grid dimensions. The sampling distribution is the mixture

$$p_{pq}(\rho) = (1-\rho)\frac{1}{G_x G_y} + \rho q_{pq}, \qquad \rho \in [0,1]. \tag{176}$$

Thus, $\rho = 0$ corresponds to uniform spatial sampling and $\rho = 1$ to fully adaptive inverse-energy sampling. The reported Darcy-flow experiments use

$$\rho = 1. \tag{177}$$

The six reported values of κ index the functional, rather than the sampling design.

Given $p_{pq}(\rho)$, we draw

$$J_{ik} \overset{\text{iid}}{\sim} \text{Categorical}\{p_{pq}(\rho)\}_{p,q}, \qquad k = 1, \dots, K, \tag{178}$$

with replacement, sort the flattened grid indices, and observe

$$Y_{ik} = u_i(X_{J_{ik}}) + \varepsilon_{ik}, \qquad \varepsilon_{ik} \sim \mathcal{N}(0, \sigma_\varepsilon^2), \qquad \sigma_\varepsilon = 0.01. \tag{179}$$

For **oracle calculations**, the inverse-design weight at grid location x_{pq} is

$$\xi_i(x_{pq}) = \frac{w_{pq}}{p_{pq}(\rho)}, \tag{180}$$

where w_{pq} is the normalized two-dimensional trapezoidal quadrature weight.

Functional	Definition	Implementation	Hyperparameters
`smooth_excess_above_threshold`	$g_{\text{exc}}(u) = \int_\Omega \frac{1}{\kappa}\log(1+\exp(\kappa^\star(u(x)-c)))\,\mathrm{d}\mu(x)$	$g_{\text{exc}}(u) = \sum_{p=1}^{H}\sum_{q=1}^{W} w_{pq}\frac{1}{\kappa}\log\big(1+\exp(\kappa(u_{pq}-c))\big)$	$c = 0.5,\ \kappa^\star = 7.5 + 2.5\kappa; \kappa \in \{0.0, 0.2, 0.4, 0.6, 0.8, 1.0\}$

Table 13: Functional-specific hyperparameters used in the reported Darcy-flow experiments.

• **Functional estimation:** The population target is the mean functional

$$\theta = \mathbb{E}[g(S_0(A))], \tag{181}$$

where the benchmark ground truth is computed from a disjoint truth pool of size 2000. The estimator itself is evaluated on the held-out test split of size 64, and all reported results are aggregated over 50 repeated runs.

For the smooth excess-above-threshold functional, the corresponding Riesz representer under the benchmark reference measure is the pointwise sigmoid field

$$w_g(u)(x) = \sigma\big(\kappa^\star(u(x) - c)\big), \qquad \sigma(z) = \frac{1}{1+e^{-z}}. \tag{182}$$

In discrete form, the same pointwise expression is evaluated on the spatial grid (see Table 13). More details on the closed-form functional derivatives and their Riesz representers are given in Supplement I.

I EXAMPLES OF FUNCTIONAL DERIVATIVES AND THEIR RIESZ REPRESENTERS

In some cases, the functional derivatives $\mathrm{D}g_u(h)$ and their Riesz representers $w_g(u)$ can be computed analytically. In the following, we summarize the functionals used in our experiments (Section 5), and provide closed-form solutions for the functional derivatives and their Riesz representers, which allows us to properly validate our results.

Note that these closed-form solutions are not always easily tractable but can involve complex derivations. Yet, DOPE is applicable even when these closed-form solutions are **not** available (see Section 4.2 and Section 4.4). In the following, all integrals are w.r.t. Lebesgue-measure μ.

• **Normalized AUC:** The normalized area under the (concentration) curve is given by

$$g(u) = (1/T) \int_0^T u(t)\,\mathrm{d}t. \tag{183}$$

The corresponding Fréchet derivative in direction h is

$$\mathrm{D}g_u(h) = (1/T) \int_0^T h(t)\,\mathrm{d}t. \tag{184}$$

Then, its Riesz representer under $\mu(\mathrm{d}t) = \mathrm{d}t/T$ is

$$w(u)(t) = 1. \tag{185}$$

For our experiments, this functional is especially interesting as it isolates the inverse-design component; w_g is constant, and therefore $\beta_0(\cdot)(\cdot) = \xi_0(\cdot)(\cdot)$.

• **Time-above-threshold:** We define a smooth version of the time-above-threshold (TAT) as

$$g(u) = (1/T) \int_0^T \sigma(\kappa(u(t) - c^\star))\,\mathrm{d}t, \tag{186}$$

where $c^\star$ is a clinically meaningful concentration threshold, $\kappa > 0$ controls the sharpness of the smooth threshold, and $\sigma(z) = 1/(1+\exp(-z))$. This is a smooth version of the time-above-threshold / time-in-range functional, since

$$\sigma(\kappa x) \xrightarrow[\kappa\to\infty]{} \mathbf{1}\{x \geq 0\}, \tag{187}$$

so, pointwise in t,

$$\sigma\big(\kappa(u(t) - c^\star)\big) \to \mathbf{1}\{u(t) \geq c^\star\}. \tag{188}$$

Therefore,

$$g_{\mathrm{tat}}(u) = \frac{1}{T} \int_0^T \sigma\big(\kappa(u(t) - c^\star)\big)\,\mathrm{d}t \to \frac{1}{T} \int_0^T \mathbf{1}\{u(t) \geq c^\star\}\,\mathrm{d}t. \tag{189}$$

The corresponding Fréchet derivative in direction h is

$$\mathrm{D}g_u(h) = (1/T) \int_0^T \kappa\sigma(\kappa(u(t) - c^\star))(1 - \sigma(\kappa(u(t) - c^\star)))h(t)\,\mathrm{d}t. \tag{190}$$

Its Riesz representer under $\mu(\mathrm{d}t) = \mathrm{d}t/T$ is

$$w_g(u)(t) = \kappa\sigma(\kappa(u(t) - c^\star))(1 - \sigma(\kappa(u(t) - c^\star))). \tag{191}$$

This functional is PK-motivated and gives a nontrivial, threshold-localized sensitivity profile.

• **Soft cmax:** We define the soft cmax functional as

$$g(u) = (1/\lambda) \log\big[(1/T) \int_0^T \exp(\lambda u(t))\,\mathrm{d}t\big], \tag{192}$$

where the temperature parameter $\lambda > 0$ controls how sharply this approximates peak concentration. This is a standard smooth proxy for peak concentration "cmax" (continuous maximum functional)

$$\mathrm{cmax}(u) \coloneqq \sup_{t\in[0,T]} u(t), \tag{193}$$

since

$$\frac{1}{\lambda}\log\left(\frac{1}{T}\int_0^T e^{\lambda u(t)}\,\mathrm{d}t\right) \longrightarrow \sup_{t\in[0,T]} u(t) \quad \text{as } \lambda\to+\infty. \tag{194}$$

Its Fréchet derivative in direction h is

$$\mathrm{D}g_u(h) = (1/T)\int_0^T [\exp(\lambda u(t))/((1/T)\int_0^T \exp(\lambda u(s))\,\mathrm{d}s)]h(t)\,\mathrm{d}t. \tag{195}$$

Equivalently, under $\mu(\mathrm{d}t) = \mathrm{d}t/T$,

$$D_g u(h) = \int_0^T w_g(u)(t)h(t)\mu(\mathrm{d}t), \tag{196}$$

with Riesz representer

$$w_g(u)(t) = \exp(\lambda u(t))/[(1/T)\int_0^T \exp(\lambda u(s))\,\mathrm{d}s]. \tag{197}$$

This functional is peak-sensitive, nonlinear, and temporally varying, while still having a closed-form derivative.

• **Smooth excess-above-threshold:** We define the smooth excess-above-threshold functional as

$$g(u) = \int_\Omega \frac{1}{\kappa}\log\big(1+\exp(\kappa^\star(u(x)-c))\big)\,\mathrm{d}\mu(x), \tag{198}$$

where the sharpness parameter $\kappa^\star > 0$ controls how rapidly the functional transitions around the threshold c. This is a standard smooth proxy for the excess-above-threshold functional

$$\mathrm{excess}_c(u) \coloneqq \int_\Omega (u(x)-c)_+\,\mathrm{d}\mu(x), \tag{199}$$

since

$$\frac{1}{\kappa^\star}\log\big(1+\exp(\kappa^\star(u(x)-c))\big) \longrightarrow (u(x)-c)_+ \quad \text{as } \kappa^\star\to+\infty \tag{200}$$

pointwise in x, and hence

$$g(u) \longrightarrow \int_\Omega (u(x)-c)_+\,\mathrm{d}\mu(x) \quad \text{as } \kappa^\star\to+\infty. \tag{201}$$

Its Fréchet derivative in direction h is

$$\mathrm{D}g_u(h) = \int_\Omega \sigma\big(\kappa^\star(u(x)-c)\big)\,h(x)\,\mathrm{d}\mu(x), \qquad \sigma(z) = \frac{1}{1+e^{-z}}. \tag{202}$$

Equivalently,

$$\mathrm{D}g_u(h) = \int_\Omega w_g(u)(x)\,h(x)\,\mathrm{d}\mu(x), \tag{203}$$

with Riesz representer

$$w_g(u)(x) = \sigma\big(\kappa^\star(u(x)-c)\big). \tag{204}$$

This functional is nonlinear and threshold-sensitive, while still admitting a closed-form derivative and a simple Riesz representer. Note that in our experiments, we use a linear transformation $\kappa^* \coloneqq f(\kappa)$ (see Supplement H.2).