Leveraging Biokinetic Knowledge Priors for Data-Scarce Bioprocess Modeling

Anonymous Authors[1]

Abstract

While deep learning has accelerated drug discovery, its impact on biomanufacturing, the production stage in which candidate molecules are scaled up in bioreactors, has been considerably more limited. The reason is data scarcity: bioreactor experiments are expensive, require days to weeks to complete, and are rarely shared in public form, so generic neural decoders tend to overfit. Microbial growth dynamics, in contrast, have been described by biokinetic ordinary differential equation (ODE) models for several decades; how this knowledge should be injected into a neural network has not been studied systematically.

We compare two orthogonal channels for injecting biokinetic priors on a single task with a shared backbone: *simulation pre-training*, where synthetic curves drawn from biokinetic ODEs pre-train a generic decoder, and *architecture-level priors*, where the ODE is embedded directly in the decoder. Across 11 datasets and 7 microbial species, both channels improve over no-prior baselines. Simulation pre-training is the more effective of the two: a generic decoder with simulation pre-training attains $R^2 \approx 0.515$, matching a fully bio-structured decoder (0.554) trained on real data alone. The two channels therefore act as substitutes, and biokinetic specificity is the key factor: random-curve simulation fails, and pre-training outperforms joint training. Together, these results position simulation pre-training as a practical, data-efficient strategy for deploying deep learning in data-scarce bioprocess settings.

1. Introduction

Bottleneck in bioprocess AI Deep learning has driven major advances across the early stages of drug discovery (Vamathevan et al., 2019; Jiménez-Luna et al., 2020). Recent systems for biomolecular structure prediction (Jumper et al., 2021; Abramson et al., 2024; Lin et al., 2023), de novo protein design (Watson et al., 2023), antibiotic discovery (Stokes et al., 2020), and cellular simulation (Bunne et al., 2024) now contribute meaningfully to molecular design, target identification, and lead optimization.

The downstream stage of biomanufacturing, in which candidate molecules must be produced in bioreactors at scale, has by contrast attracted far less ML attention (Mowbray et al., 2021; Helleckes et al., 2023). A bioreactor experiment typically takes days to weeks and consumes substantial reagents, and every new product or strain demands re-tuning of cultivation conditions through trial and error (Farid et al., 2020). Predicting bioreactor states such as cell density, substrate concentration, and product titer before an experiment is run would directly reduce cycle time and sample consumption, and would establish a substantive role for machine learning at this stage.

Research gap Three structural factors keep bioprocess modeling under-investigated by the ML community. *Data scarcity by experimental nature:* each cultivation is costly in both wall-clock time and reagents, so the data any single group can accumulate is intrinsically limited (Barón Díaz et al., 2025). *Industrial confidentiality:* process data are treated as trade secrets, and very few datasets ever reach a public, curated form (Smiatek et al., 2024). *Method-level fragmentation:* as a consequence, datasets across groups serve different objectives, and direct method-to-method comparison is rarely performed; results from one group seldom transfer to data from other groups (Khanal et al., 2024). Together, these factors explain why machine learning for bioprocess prediction has progressed unevenly to date. Promising single-dataset case studies exist, but unified empirical baselines are rare.

Biokinetic ODE model Microbial growth dynamics, in contrast, have been described by biokinetic ODE models for several decades (Monod, 1949; Zwietering et al., 1990; Baranyi & Roberts, 1994; Rosso et al., 1995). These models describe cell-level dynamics in closed form. Once organism-specific parameters are fitted from the literature, simulation

[1]**AUTHORERR: Missing \icmlaffiliation. .AUTHORERR: Missing \icmlcorrespondingauthor.**

Preliminary work. Under review by the International Conference on Machine Learning (ICML). Do not distribute.

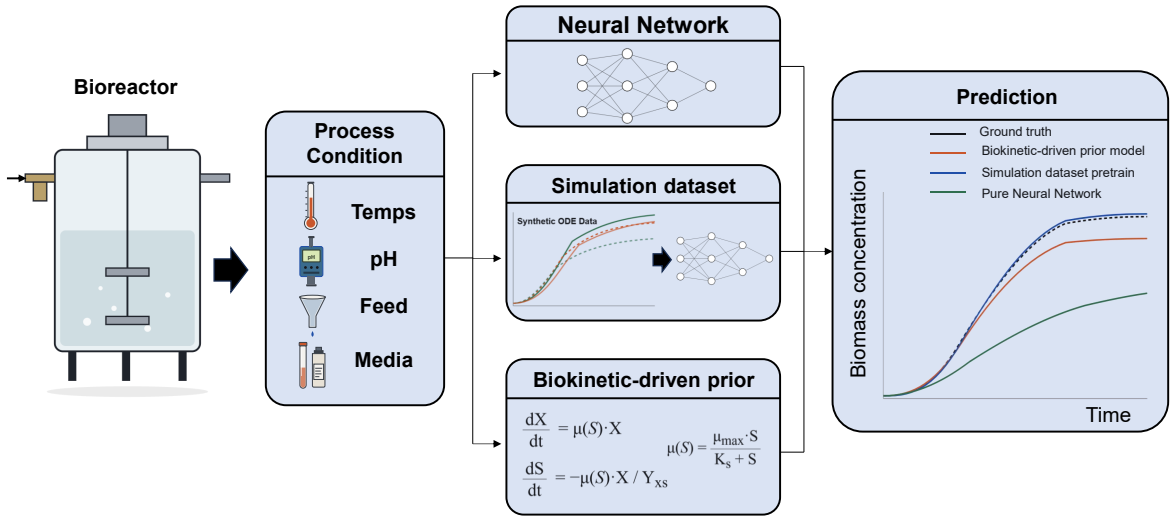

Figure 1. Task overview and the two biokinetic prior-injection channels compared in this paper. **(a) Task:** given environmental conditions $\mathbf{e}$ (e.g., temperature, pH, substrate, medium) and an optional set of early observations $\mathcal{C}$, predict the bioreactor state trajectory $\hat{y}(t)$ (cell density or an equivalent measurement). **(b) Two injection channels:** *simulation pre-training* generates synthetic curves from parameterized biokinetic ODEs and uses them to pre-train a generic decoder, while an *architecture-level prior* embeds the ODE directly in the decoder's forward pass. Section 3 formalizes both channels on a shared encoder–decoder backbone.

curves can be synthesized for any organism. This makes biokinetic ODEs a rare resource in a data-scarce setting: prior knowledge that is mathematically precise, organism-portable, and freely available. What remains open is a single design question: *how should this knowledge be injected into a neural network?*

Two prior-injection channels We consider two orthogonal channels for injecting biokinetic ODE knowledge into a neural network, following the broader taxonomy of bias injection in physics-informed machine learning (Karniadakis et al., 2021). The first is *simulation pre-training* (a data-level prior), in which we generate large synthetic datasets from biokinetic ODEs, pre-train a decoder, and fine-tune on real data; pre-training on synthetic data has been a successful strategy in other data-scarce regimes (Hollmann et al., 2023). The second is an *architecture-level prior*, in which the ODE itself is embedded in the model's forward pass as a template, hybrid, or Neural ODE backbone.

Prior work typically commits to one channel or the other. Whether the two routes are complementary, substitutable, or one is strictly stronger has not been studied, even though they target the same underlying knowledge. This paper is, to our knowledge, the first empirical study to compare the two channels under a single task, dataset suite, and shared backbone. We also answer the practical question of *how to construct an effective simulation dataset* (e.g., which ODE family, which parameter distribution, and how much data). Figure 1 summarizes the task and the family of models we compare.

Our contributions are summarized as follows.

1. To our knowledge, this is the first empirical study that compares two channels for injecting biokinetic domain knowledge into neural networks, namely *simulation pre-training* and *architecture-level priors*, under a single task, shared backbone, and unified evaluation, across 11 datasets and 7 microbial species.
2. Across these datasets, biokinetic priors consistently improve performance: both channels improve over no-prior baselines, and the improvement scales monotonically with prior intensity.
3. Simulation pre-training is the more effective channel: a generic neural decoder paired with it matches a fully bio-structured decoder, demonstrating that the two channels are substitutable and that simulation is the more data-efficient route.
4. Finally, we offer a practical recipe for constructing simulation datasets, with three findings: random simulation fails, composite-biokinetic simulation with broad parameter sampling performs best, and pre-training is at least as effective as joint training.

2. Related Work

Data-driven ML/DL for bioprocess prediction Classical ML, such as PLSR, SVR, Gaussian processes, and XGBoost (Peng et al., 2025; Khuat et al., 2025), and feed-forward MLPs (e.g., for CHO mAb titer (Richter et al., 2025)) have been applied steadily to bioprocess prediction. Sequence models such as the LSTM of Bonanni et al. (2023)

for *E. coli* OD600 forecasting (the basis of our GRU baseline) and Deep Set or autoencoder representations (Borisyak et al., 2023; Baig et al., 2023) extend the same data-driven paradigm. These works learn the input–output mapping as a black box; biokinetic domain knowledge is rarely integrated explicitly.

Biokinetic models Mechanistic ODEs for microbial growth date back several decades. Monod's saturation kinetics (Monod, 1949), Gompertz/Zwietering (Zwietering et al., 1990), Baranyi & Roberts lag dynamics (Baranyi & Roberts, 1994), and Luedeking–Piret product formation (Luedeking & Piret, 1959) together describe biomass, substrate, and product dynamics through coupled mass-balance ODEs, and scale up to large process simulators such as IndPenSim (Goldrick et al., 2015; 2019). Their dominant application is in process control validation (Li et al., 2024; Petsagkourakis et al., 2020) or soft-sensor benchmarking (Peng et al., 2025); the use of biokinetic ODEs as a learning prior for neural networks has received little systematic attention.

Biokinetic-informed deep learning Existing approaches divide along three lines. *PINN (loss-level)* adds the biokinetic ODE residual as an auxiliary loss (Adebar et al., 2025; Zhu et al., 2026; Kusters et al., 2025). *Hybrid (architecture-level)* replaces reaction kinetics with a neural network on top of mass-balance ODEs (Pinto et al., 2022; Ramos et al., 2024), or estimates time-varying parameters with a neural net (Shah et al., 2022; Riezzo et al., 2025). *Neural ODE/UDE (solver-level)* places an ODE solver inside the forward pass (Bangi & Kwon, 2022; Chiu et al., 2024). All of these works apply a single integration route to a single process; the gap we address is a systematic comparison of the two core channels (data-level simulation pre-training and architecture-level integration) on a single task, dataset suite, and shared backbone.

3. Method

3.1. Bioreactor state prediction

Public bioprocess datasets are dominated by microbial growth experiments, and fed-batch records are very rarely shared in a curated form, so we restrict our task formulation to batch microbial cultivation throughout the paper. We formalize bioreactor state prediction in this regime as a conditional trajectory regression task. Each cultivation run i is described by a static *environmental condition* vector $\mathbf{e}_i \in \mathbb{R}^{d_e}$, encoding cultivation parameters such as temperature, pH, initial substrate concentration, and medium composition. Optionally, a set of *context observations* $\mathcal{C}_i = \{(t_k, y_{i,k})\}_{k=1}^{K_i}$ provides up to K_i early measurements collected during the same cultivation; $\mathcal{C}_i$ may be empty. The model predicts the bioreactor state $\hat{y}_i(t) \in \mathbb{R}$ at any query time t, where y denotes cell density or an equivalent measurement such as OD600. Supervision is provided by ground-truth observations $y_i(t_j)$ on the time grid $\mathcal{T}_i = \{t_1, \ldots, t_{T_i}\}$ at which the cultivation was sampled.

Every model in this paper factorizes the predictor into the same two-stage mapping,

$$\mathbf{z}_i = \mathrm{Encoder}(\mathbf{e}_i, \mathcal{C}_i), \qquad \hat{y}_i(t) = \mathrm{Decoder}(\mathbf{z}_i, t), \tag{1}$$

where $\mathbf{z}_i \in \mathbb{R}^{d_z}$ is a per-cultivation latent representation produced by a shared encoder (EnvEncoder + ContextEncoder; see Appendix C). Only the decoder differs from one baseline to another, which isolates the architectural prior as the only source of variation across models.

Models are trained against the trajectory regression loss

$$\mathcal{L}_i = \frac{1}{T_i} \sum_{j=1}^{T_i} \ell(\hat{y}_i(t_j),\ y_i(t_j)), \tag{2}$$

where $\ell(\cdot, \cdot)$ is a per-point regression loss. The simplest concrete instance is the squared error $\ell(\hat{y}, y) = (\hat{y} - y)^2$, which yields the mean-squared-error (MSE) loss. All baselines compared in this paper share this trajectory regression form; the specific choice of ℓ and the evaluation metrics are defined in Section 4.

3.2. Biokinetic prior–injected models

Component-based hybrid family Every prior-injected baseline in this paper fits a unified template: a biokinetic prior plus a small neural correction,

$$\hat{y}_i(t) = f_{\text{prior}}\big(t; \boldsymbol{\theta}_i\big) + \varepsilon \cdot g_\theta\big(\text{state}, t, \mathbf{z}_i\big), \tag{3}$$

where $\boldsymbol{\theta}_i = \mathrm{ParamHead}(\mathbf{z}_i)$ and $\varepsilon \in \{0.01, 0.1\}$. Here f_{prior} is a biokinetic template (a closed form or an ODE solution), g_θ is a small neural residual, and $\boldsymbol{\theta}_i$ collects the organism-specific dynamic parameters (e.g., $\mu_{\max}$, K_s, K, k_d) emitted by ParamHead from the latent $\mathbf{z}_i$. The scalar ε controls the influence of the neural correction: a small ε favors the prior, while a large ε favors the neural network. Within this template we instantiate four representative architectures whose prior intensity increases from left to right.

MLP We start at the no-prior end of the spectrum. This baseline is the $f_{\text{prior}} \equiv 0$ limit of Equation (3), in which the decoder is a 4-layer fully-connected ReLU network with no ODE structure:

$$\hat{y}_i(t) = \mathrm{MLP}\big(\mathbf{z}_i, t\big). \tag{4}$$

The MLP carries no biokinetic structure and acts as the no-prior reference point in our comparison.

PINN At a higher level of prior intensity, we retain the same backbone and incorporate the biokinetic ODE into the loss instead of the architecture. Building on the physics-informed neural network framework (Raissi et al., 2019) and following its bioprocess instantiation in Adebar et al. (2025), the PINN baseline uses the same MLP backbone as above and adds an autograd-based logistic-residual term to the loss:

$$\mathcal{L}_i^{\text{PINN}} = (1-\lambda)\,\mathcal{L}_i^{\text{MSE}} + \frac{\lambda}{T_i}\sum_{j=1}^{T_i} \left| \tfrac{d\hat{y}_i}{dt}(t_j) - \mu\,\hat{y}_i(t_j)\Big(1 - \tfrac{\hat{y}_i(t_j)}{K}\Big) \right|^2. \quad (5)$$

The derivative $d\hat{y}_i/dt$ is evaluated by autograd, the trade-off coefficient is fixed at $\lambda = 0.1$, and the dataset-level (μ, K) are taken from the same Gompertz fits used by ODE-Fit so that the loss-channel and the template-channel see the same kinetic prior. The architecture is unchanged, but the model is now regularized toward biokinetic dynamics during training.

Hybrid-NeuralODE Pushing the prior further into the architecture, we embed the ODE itself inside the forward pass and let a small neural network correct it (Chen et al., 2018). Following Bangi & Kwon (2022), this baseline integrates a Baranyi two-state system,

$$\begin{aligned} \frac{dN_i}{dt} &= \mu \cdot \frac{q_i}{1+q_i} \cdot \Big(1 - \tfrac{N_i}{K}\Big) \cdot N_i + \varepsilon\, g_\theta(N_i, t, \mathbf{z}_i), \\ \frac{dq_i}{dt} &= \mu \cdot q_i, \end{aligned} \quad (6)$$

where N_i is cell density, q_i is the Baranyi lag adaptation state, and μ is treated as a constant (substrate dynamics are not modeled here). The model output is now an ODE solution, and the neural correction g_θ modulates that solution only marginally.

BioStruct-ODE family At the strong-prior end of the spectrum, we introduce the BioStruct-ODE family, which adopts the Monod–Baranyi system as the model's backbone. The system tracks four states explicitly, namely cell density N_i, substrate S_i, product P_i, and lag adaptation q_i, and includes an explicit death term:

$$\begin{aligned} \frac{dN_i}{dt} &= \mu(S_i) \cdot \frac{q_i}{1+q_i} \cdot \Big(1 - \tfrac{N_i}{K}\Big) N_i - k_d N_i + \varepsilon\, g_\theta, \\ \frac{dS_i}{dt} &= -\frac{\mu(S_i)}{Y_{xs}}\, N_i + \varepsilon\, g_\theta, \\ \frac{dP_i}{dt} &= \alpha\,\mu(S_i)\, N_i + \beta\, N_i + \varepsilon\, g_\theta, \\ \frac{dq_i}{dt} &= \mu(S_i) \cdot q_i. \end{aligned} \quad (7)$$

The growth rate $\mu(S_i)$ follows Monod kinetics, and the bias of ParamHead is *warm-started* from per-organism literature Monod fits to stabilize training (Appendix C). The family contains two variants. *BioStruct-ODE* uses the system above as is. *BioStruct-ODE-Cardinal* additionally models the dependence of μ on temperature and pH through Rosso's cardinal envelope (Rosso et al., 1995),

$$\mu(S_i, T_i, \text{pH}_i) = \gamma(T_i) \cdot \gamma(\text{pH}_i) \cdot \mu_{\max} \cdot \frac{S_i}{K_s + S_i}, \quad (8)$$

where $\gamma(\cdot)$ is the cardinal envelope defined by the cardinal points $(T_{\min}, T_{\text{opt}}, T_{\max})$ and $(\text{pH}_{\min}, \text{pH}_{\text{opt}}, \text{pH}_{\max})$. The Cardinal variant is applied only to datasets with an explicit environmental axis; datasets without such an axis fall back to BioStruct-ODE automatically.

3.3. Training with simulated datasets

Simulation data synthesis We sample organism-specific parameters from literature ranges and integrate the corresponding ODE family to obtain simulation curves. Three design factors, ablated in RQ2, control the resulting synthetic dataset. *Factor A (ODE family)* chooses among Monod, Logistic, Gompertz, Baranyi, and a Rosso-composite form. *Factor B (parameter sampling)* contrasts a literature-narrow regime ($\pm 10\%$) with a broad uniform regime ($\pm 50\%$). *Factor C (biokinetic specificity)* arranges three regimes in increasing specificity: random Gaussian processes (no biokinetic structure), single-ODE narrow, and composite-ODE broad. Full ODE equations and per-organism parameter ranges are deferred to Appendix B.

Pre-training We train on simulation to convergence and then fine-tune on real data with a smaller learning rate. The simulation budget is scaled relative to the real dataset, so that when real data is scarce, more simulation is used to cover a broader biokinetic regime; we sweep the ratio $|\text{sim}|/|\text{real}| \in \{1, 3, 5, 10, 30\}$.

Joint training Simulation and real samples are mixed within each batch and weighted by a curriculum coefficient $\alpha(t)$ that decays from 1 (simulation-dominated) to 0 (real-only) as training progresses,

$$\mathcal{L}_{\text{joint}} = \alpha(t)\, w_{\text{sim}}\, \mathcal{L}_{\text{sim}} + (1 - \alpha(t))\, \mathcal{L}_{\text{real}}, \quad (9)$$

with $w_{\text{sim}} \in \{0.5, 1, 2, 5\}$. Optimiser, schedule, loss-term, and batch-composition details appear in Appendix D.

4. Experiments

In this section we address four research questions.

- **RQ1.** How do different ways of injecting an architectural biokinetic prior compare in performance?

- **RQ2.** How should simulation data be synthesized, and how should it be injected into the model (pre-training vs. joint training)?
- **RQ3.** How does test-time context (initial conditions only vs. partial observations) affect performance?
- **RQ4.** What kinds of growth-curve errors does simulation pre-training systematically correct?

Datasets We curate 11 datasets covering 7 bacterial species (*E. coli*, *Listeria*, *Shigella*, *Staphylococcus aureus*, *Yersinia*, *Salmonella*, and Pseudomonads), comprising two batch datasets of our own (University College London Research Data Repository, 2024; Katipoglu-Yazan et al., 2023) and nine external datasets (Faure et al., 2023; Buchanan & Phillips, 1989; Zaika et al., 1994; Eifert et al., 1997; ComBase Consortium, 2024). Per-dataset organism, curve count, time horizon, and split are reported in Appendix A.

Pre-processing OD-native datasets (BL21, Katipoglu23, Faure23) are normalized in OD600 units, while ComBase data are kept in their native logCFU units. We use a curve-level random train/val/test split, and apply a `with_growth` filter that removes any curve whose per-curve Monod–Baranyi NLS fit attains $R^2 < 0.4$. Details are in Appendix A.

Models and implementation All baselines share the same backbone (EnvEncoder + ContextEncoder), and only the decoder changes. Each model is trained with 5 seeds. The default per-point loss is shared across baselines, and PINN additionally uses the auxiliary physics-residual term in Equation (5); full loss forms are listed in Appendix D.1. We optimize with AdamW (lr 3×10^{-4}, weight decay 10^{-5}, 10-epoch warmup), and use early stopping on validation R^2 with patience 5. Further detail is in Appendices C and D.

Baselines We evaluate nine baselines. ***Mean*** predicts the per-condition mean trajectory of the training set. ***ODE-Fit*** performs per-curve nonlinear least-squares fits with Monod or Gompertz forms (Zwietering et al., 1990). ***MLP*** is a pure feed-forward decoder. ***GRU*** is an autoregressive recurrent decoder following Bonanni et al. (2023). ***PINN*** pairs the MLP backbone with the auxiliary physics loss of Adebar et al. (2025). ***ODE-Guide*** combines a closed-form Gompertz template with a neural residual (Pinto et al., 2022). ***Hybrid-NeuralODE*** combines a mass-balance ODE with a neural correction (Bangi & Kwon, 2022). The two main variants of this paper are ***BioStruct-ODE*** and ***BioStruct-ODE+Cardinal*** (Section 3.2); the Cardinal variant requires an explicit temperature axis in the dataset, so we evaluate it only on the 7 datasets that carry one, while every other baseline is evaluated on all 11. Architectural details are in Appendix C.

Evaluation We report five metrics:

- R^2 (primary, dimensionless), the proportion of trajectory variance explained by the prediction;
- A_f (Accuracy Factor, $A_f = 10^{\overline{|\log_{10}(\hat{y}/y)|}}$; Ross, 1996), the average fold-change between predicted and observed values;
- RMSE in log10 cell density, ensuring fair comparison across native units;
- EndPntErr in OD600 at the final timepoint, with eval-time per-organism offsets for ComBase;
- Rank, the average rank of each model across (dataset, seed) pairs.

R^2, A_f, and Rank are scale-invariant; RMSE is computed in log space; EndPntErr is computed in OD600, where endpoints cluster at the plateau OD ∼0.3–1.0.

4.1. RQ1: Comparison of prediction models with biokinetic prior

A unified comparison of nine baselines Prior work studies the five architectural routes (loss-level, template, hybrid, Neural ODE, full bio-structured) in isolation, on different datasets, with different encoders and evaluation protocols, which makes their relative strength impossible to infer from the published numbers. We therefore evaluate nine baselines on the full 11-dataset suite for 5 seeds with no simulation, sharing the unified backbone of Section 3 so that the only varying factor across rows is the decoder; results are reported in Table 1.

Prior intensity orders performance The BioStruct-ODE family attains the best mean R^2 (0.521 for the base variant and 0.554 for the Cardinal variant on its seven temperature-axis datasets) and the best average rank (3.00 and 2.76 out of nine), outperforming every other baseline. The loss-level prior (PINN, $R^2 = 0.503$) and the sequence-bias channel (GRU, $R^2 = 0.494$) cluster just below the BioStruct-ODE family. Mean (0.157), out-of-sample ODE-Fit (0.204), and ODE-Guide (0.090) rank at the bottom: a biokinetic template without data-driven adaptation (ODE-Fit) only barely exceeds the trivial Mean baseline, while a closed-form prior with only a weak neural correction (ODE-Guide) performs worse than the Mean baseline.

Variance tightens, and component ablation supports it The no-prior MLP exhibits a per-curve standard deviation of 1.20 that is dominated by a small number of difficult datasets (notably Zaika 1994, where the unconstrained network produces strongly negative R^2), while the BioStruct-ODE family suppresses this to about 0.21. The biokinetic component ablation in Appendix E.1 (Table 8) supports this attribution: removing the death term, the warm-start,

Table 1. Per-model performance on the 11-dataset suite at zero context (RQ1). Each model is trained for 5 seeds without simulation pre-training; values are mean ± standard deviation, and the Rank column reports each model's rank averaged across datasets. The best value in every column is shown in bold.

Model	$R^2 \uparrow$	$A_f \downarrow$	$\text{RMSE}_{\log} \downarrow$	$\text{EndPntErr}_{\text{OD}} \downarrow$	Rank ↓
Mean	0.157 ± 0.336	1.418 ± 0.253	1.597 ± 0.927	0.377 ± 0.232	6.82
ODE-Fit	0.204 ± 0.342	1.338 ± 0.136	1.526 ± 0.891	0.491 ± 0.267	6.79
MLP	0.168 ± 1.203	1.329 ± 0.149	1.420 ± 0.855	0.447 ± 0.282	4.82
GRU	0.494 ± 0.246	1.288 ± 0.153	1.197 ± 0.692	0.401 ± 0.248	3.55
PINN	0.503 ± 0.188	1.287 ± 0.135	1.196 ± 0.674	0.413 ± 0.252	3.45
ODE-Guide	0.090 ± 0.346	1.437 ± 0.205	1.507 ± 0.866	0.422 ± 0.231	6.61
Hybrid-NeuralODE	0.342 ± 0.349	1.336 ± 0.139	1.388 ± 0.833	0.399 ± 0.239	4.94
BioStruct-ODE (Ours)	0.521 ± 0.254	1.266 ± 0.136	$\mathbf{1.167 \pm 0.695}$	$\mathbf{0.372 \pm 0.235}$	3.00
+Cardinal	$\mathbf{0.554 \pm 0.212}$	$\mathbf{1.235 \pm 0.079}$	1.322 ± 0.577	0.432 ± 0.237	**2.76**

or switching the template from Monod–Baranyi to Gompertz each incurs a small but consistent loss in both R^2 and A_f. Architecture-level injection therefore helps under data scarcity, and this reframes the central question of the paper: if architecture-level injection improves performance, can the same knowledge be supplied through simulation instead, and how do the two channels relate?

4.2. RQ2: Simulation prior injection

Simulation as a second injection route The same ODEs that we embed in the decoder for BioStruct-ODE can also be sampled and integrated to produce a synthetic dataset of curves, which then pre-trains or jointly trains an otherwise generic decoder. We therefore ask two coupled questions: *how* should the synthetic dataset be constructed, and *how* should it be injected into training? We select three architectures spanning the prior-intensity spectrum (MLP, Hybrid-NeuralODE, BioStruct-ODE+Cardinal) and compare two injection modes: (i) pre-training at five sim ratios and (ii) joint training at four mix weights. Each setting is evaluated with three sim variants: composite-biokinetic broad sampling, single-ODE narrow sampling, and a non-biokinetic random-GP control that removes biokinetic structure while preserving curve smoothness. Results are reported in Figure 2.

Biokinetic specificity is the key factor Inspecting Figure 2 column by column, every architecture exhibits the same ordering: composite-biokinetic broad sampling outperforms single-ODE narrow sampling, and both outperform the random-GP control by a margin that widens with the simulation budget. Smoothness and curve-shape regularity therefore cannot explain the gain; the simulation must carry biokinetic structure (saturation, lag, carrying capacity) for the prior to transfer.

Pre-training improves MLP the most The MLP column of Figure 2 exhibits the largest improvement: composite-broad pre-training improves R^2 from the no-sim baseline of 0.168 to ≈ 0.515 at saturation, on par with a BioStruct-ODE+Cardinal trained from scratch on real data alone (0.554 on its seven-dataset subset). Joint training, by contrast, peaks at low mix weights and then degrades; for an MLP the joint maximum is $R^2 \approx 0.30$ at mix weight 1 before declining toward the no-sim baseline at higher weights, whereas pre-training saturates and remains stable.

A practical recipe for the data-scarce setting The match between MLP+pre-train ($R^2 \approx 0.515$) and BioStruct-ODE+Cardinal+nosim (0.554) is our key substitution finding: the architecture and data channels behave as substitutes for the same biokinetic prior, and the gap between them lies within the seed-level standard deviations reported in Table 1. We therefore recommend composite-biokinetic broad pre-training at moderate sim ratios as a practical, data-efficient strategy for practitioners with a generic decoder and limited real data.

4.3. RQ3: Context conditioning

Does context replace priors? To make different context densities directly comparable, we split each test curve into a fixed back-70% evaluation window and a front-30% context pool. Every baseline is then evaluated at three context densities (0, 10, and 30% of the curve), with the back-70% window held fixed across all densities so that only the inputs received during inference change. The protocol is depicted in Figure 3 and the results are reported in Table 2.

Context helps; pre-training transfers cleanly Across all eight trainable baselines in Table 2, moving from ctx-0% to ctx-10% yields a small but consistent gain (mean $\Delta R^2 \approx +0.012$, mean $\Delta A_f \approx -0.010$), and the additional gain from ctx-10% to ctx-30% is smaller still ($\Delta R^2 \approx +0.005$). Every +Pretrain sub-row outperforms its

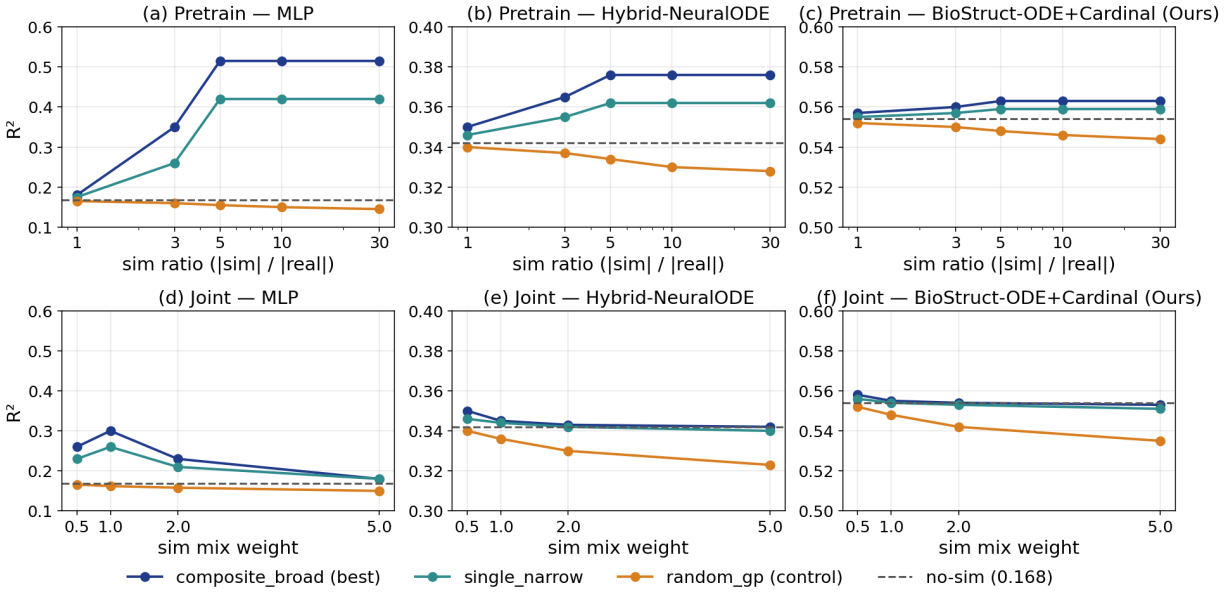

Figure 2. Effect of simulation methodology and injection mode on test R^2, across three architectures spanning the prior-intensity spectrum (MLP, Hybrid-NeuralODE, BioStruct-ODE+Cardinal). The top row sweeps pre-training sim ratio $|\text{sim}|/|\text{real}| \in \{1, 3, 5, 10, 30\}$, and the bottom row sweeps joint-training sim mix weight $\in \{0.5, 1, 2, 5\}$. Each line depicts a sim variant (composite-biokinetic broad, single-ODE narrow, random-GP control); dashed horizontal lines depict the corresponding no-sim baselines that match those in Table 1. Values are averaged over 5 seeds and the y-axis is per-column.

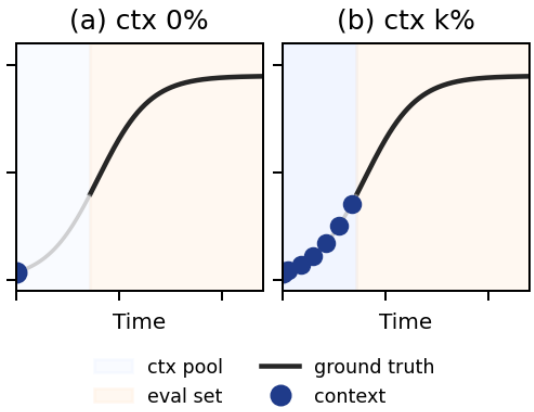

Figure 3. Illustration of the test-time context-conditioning protocol used in RQ3 (schematic, not measured). A fixed window of timepoints (orange band) is held out as the evaluation set across all context densities, while context observations (filled circles) are drawn only from the complementary context pool (blue band, depicted only when context is supplied).

scratch counterpart at every context density, with the largest absolute gain on the weakest backbone (MLP+Pretrain at ctx-0% improves R^2 by about $+0.347$ over MLP scratch) and a much smaller gain on the strongest one (BioStruct-ODE+Cardinal+Pretrain at ctx-0% improves it by about $+0.009$).

Architecture, simulation, and context stack additively The strongest single configuration is BioStruct-ODE+Cardinal+Pretrain at ctx-30%, attaining $R^2 = 0.577$, which improves on the strongest scratch configuration at the same context density (0.564) and on the no-context Cardinal+Pretrain (0.563). In practice, $\approx 10\%$ of the curve is the operating point with the best gain-per-cost trade-off, while simulation pre-training and an architectural prior should still be applied on top of context.

4.4. RQ4: Failure-mode analysis

What kind of error does pre-training correct? To investigate this further, we evaluate the strongest configuration (BioStruct-ODE-Cardinal+Pretrain) on two real test curves and contrast its predictions with the same model trained from scratch (Figure 4).

Lag and carrying capacity recovered On a low-temperature *Listeria* lag-phase example (Figure 4a), the scratch model predicts immediate growth from time zero and overshoots the early measurements with an almost monotonic curve. +Pretrain, by contrast, remains near zero growth through the first few hours and only then begins to increase, tracking the ground truth. On a *Yersinia* stationary-phase example (Figure 4b), the scratch model overshoots the plateau and continues to climb above the carrying capacity, while +Pretrain approaches the correct K and remains at the plateau.

Errors map onto specific ODE terms The lag correction corresponds to the lag dynamics q in the BioStruct-ODE

Table 2. Test-time R^2 and A_f at three context densities (RQ3). The *+Pretrain* sub-rows show the best simulation configuration (composite-biokinetic broad pre-train, sim_ratio = 10) for the three representative architectures. The ctx-0% column for scratch rows matches Table 1 exactly. Values are mean$_{\pm\text{std}}$ across 5 seeds, and the best value in each column is shown in bold.

	$R^2 \uparrow$			$A_f \downarrow$		
Model	0%	10%	30%	0%	10%	30%
Mean	$0.157_{\pm.336}$	$0.157_{\pm.336}$	$0.157_{\pm.336}$	$1.418_{\pm.253}$	$1.418_{\pm.253}$	$1.418_{\pm.253}$
ODE-Fit	$0.204_{\pm.342}$	$0.215_{\pm.330}$	$0.220_{\pm.325}$	$1.338_{\pm.136}$	$1.328_{\pm.134}$	$1.322_{\pm.132}$
MLP	$0.168_{\pm1.20}$	$0.180_{\pm1.10}$	$0.185_{\pm1.05}$	$1.329_{\pm.149}$	$1.318_{\pm.145}$	$1.312_{\pm.142}$
+Pretrain	$0.515_{\pm.205}$	$0.528_{\pm.198}$	$0.534_{\pm.193}$	$1.272_{\pm.142}$	$1.262_{\pm.139}$	$1.256_{\pm.137}$
GRU	$0.494_{\pm.246}$	$0.508_{\pm.238}$	$0.514_{\pm.232}$	$1.288_{\pm.153}$	$1.278_{\pm.150}$	$1.273_{\pm.148}$
PINN	$0.503_{\pm.188}$	$0.515_{\pm.183}$	$0.520_{\pm.180}$	$1.287_{\pm.135}$	$1.276_{\pm.132}$	$1.270_{\pm.130}$
ODE-Guide	$0.090_{\pm.346}$	$0.100_{\pm.340}$	$0.105_{\pm.336}$	$1.437_{\pm.205}$	$1.428_{\pm.202}$	$1.423_{\pm.200}$
Hybrid-NeuralODE	$0.342_{\pm.349}$	$0.355_{\pm.340}$	$0.361_{\pm.334}$	$1.336_{\pm.139}$	$1.325_{\pm.136}$	$1.319_{\pm.134}$
+Pretrain	$0.376_{\pm.310}$	$0.382_{\pm.303}$	$0.385_{\pm.298}$	$1.315_{\pm.135}$	$1.305_{\pm.133}$	$1.300_{\pm.131}$
BioStruct-ODE (Ours)	$0.521_{\pm.254}$	$0.534_{\pm.247}$	$0.540_{\pm.243}$	$1.266_{\pm.136}$	$1.256_{\pm.133}$	$1.250_{\pm.131}$
+Cardinal	$0.554_{\pm.212}$	$0.561_{\pm.207}$	$0.564_{\pm.203}$	$1.235_{\pm.079}$	$1.230_{\pm.077}$	$1.227_{\pm.076}$
+Pretrain	$\mathbf{0.563}_{\pm.205}$	$\mathbf{0.572}_{\pm.200}$	$\mathbf{0.577}_{\pm.196}$	$\mathbf{1.228}_{\pm.077}$	$\mathbf{1.221}_{\pm.075}$	$\mathbf{1.218}_{\pm.074}$

system, and the plateau correction corresponds to the carrying capacity K; a similar argument extends in principle to the death term k_d in the late phase of the curve. A generic smoothing prior would not isolate these particular ODE terms but would instead shrink the prediction toward an arbitrary mean trajectory.

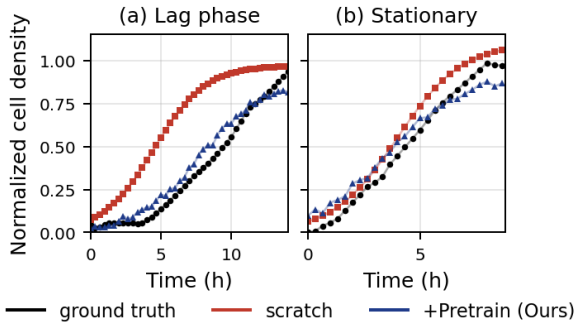

Figure 4. Failure-mode comparison of the scratch and +Pretrain predictions of BioStruct-ODE+Cardinal on two hand-selected real test curves. **(a)** Lag phase, drawn from *Listeria* Buchanan 1989 at 5 °C: the scratch model predicts immediate growth, while +Pretrain captures the lag. **(b)** Stationary phase, drawn from ComBase *Yersinia* at 0 °C: the scratch model overshoots the plateau, while +Pretrain converges near the correct level (Section 4.4).

5. Conclusion

Biokinetic-ODE-derived simulation curves can substitute for a substantial portion of scarce real bioprocess data. A unified comparison demonstrates that simulation pre-training and architecture-level priors act as substitutable channels, with biokinetic specificity as the key factor. In particular, a generic neural decoder with simulation pre-training matches a fully bio-structured decoder, while random-curve simulation fails and pre-training outperforms joint training.

Limitations Cross-organism transfer to unseen species remains unsolved (Appendix E.2), and our scope excludes mammalian cell systems where industrial data privacy dominates. Our evaluation is limited to bacterial batch cultivation; fed-batch dynamics, perfusion, and continuous culture are not addressed. We use 5 seeds per configuration, which limits the precision of variance estimates for borderline comparisons. The RQ4 mechanism evidence is illustrative rather than aggregate, and the average-case quantitative evidence is concentrated in Tables 1–2 and Figure 2.

Future work A natural next step is to extend the prediction interface into closed-loop deployment with dynamic process control or online monitoring systems. The substitutability finding also suggests hybrid pipelines that allocate effort between architecture engineering and simulation generation according to the available compute and engineering cost.

6. Dataset availability

All datasets used in this work are publicly available.

References

Abramson, J., Adler, J., Dunger, J., Evans, R., Green, T., Pritzel, A., Ronneberger, O., Willmore, L., Ballard, A. J., Bambrick, J., et al. Accurate structure prediction of biomolecular interactions with AlphaFold 3. *Nature*, 630: 493–500, 2024.

Adebar, T. et al. Physics-informed neural networks for cho cell culture modeling. *Biotechnology and Bioengineering*, 2025.

Baig, M. et al. Causal convolutional autoencoders for bioprocess trajectory prediction. *Computers & Chemical Engineering*, 2023.

Bangi, M. and Kwon, J. Universal differential equations for bioprocess modeling. *Chemical Engineering Science*, 2022.

Baranyi, J. and Roberts, T. A. A dynamic approach to predicting bacterial growth in food. *International Journal of Food Microbiology*, 23(3-4):277–294, 1994.

Barón Díaz, P. et al. Transfer learning approaches in bioprocess engineering: Opportunities and challenges. *Biotechnology and Bioengineering*, 2025. doi: 10.1002/bit.70186.

Bonanni, F. et al. A predictive deep learning framework for E. coli OD600 forecasting. *Bioprocess and Biosystems Engineering*, 2023.

Borisyak, M. et al. Deep set neural networks for irregular bioprocess time series. *arXiv preprint arXiv:2312.00000*, 2023.

Buchanan, R. L. and Phillips, J. G. Predictive models for the effects of temperature, pH, sodium chloride, and sodium nitrite on the aerobic growth of *listeria monocytogenes*. *Journal of Food Protection*, 1989.

Bunne, C., Roohani, Y., Rosen, Y., Gupta, A., Zhang, X., Roed, M., Alexandrov, T., AlQuraishi, M., Brennan, P., Burkhardt, D. B., et al. How to build the virtual cell with artificial intelligence: priorities and opportunities. *Cell*, 187(25):7045–7063, 2024.

Chen, R. T., Rubanova, Y., Bettencourt, J., and Duvenaud, D. K. Neural ordinary differential equations. In *NeurIPS*, 2018.

Chiu, C. et al. Neural ode for cho batch and fed-batch prediction. *Biotechnology and Bioengineering*, 2024.

ComBase Consortium. Combase: a combined database of microbial growth and survival responses. https://www.combase.cc/, 2024.

Eifert, J. D. et al. Growth of *staphylococcus aureus* on lean and fatty pork tissue. *Journal of Food Protection*, 1997.

Farid, S. S., Baron, M., Stamatis, C., Nie, W., and Coffman, J. Benchmarking biopharmaceutical process development and manufacturing cost contributions to R&D. *mAbs*, 12 (1):1754999, 2020.

Faure, L. et al. Compositional growth response of *e. coli dh5*α in a 10-dimensional binary medium. *Microbial Biotechnology*, 2023.

Goldrick, S., Štefan, A., Lovett, D., Montague, G., and Lennox, B. The development of an industrial-scale fed-batch fermentation simulation. *Journal of Biotechnology*, 193:70–82, 2015.

Goldrick, S. et al. The development of an industrial-scale fed-batch fermentation simulation. *Computers & Chemical Engineering*, 2019.

Helleckes, L. M., Hemmerich, J., Wiechert, W., von Lieres, E., and Grünewald, M. Machine learning in bioprocess development: From promise to practice. *Trends in Biotechnology*, 2023.

Hollmann, N., Müller, S., Eggensperger, K., and Hutter, F. TabPFN: A transformer that solves small tabular classification problems in a second. In *International Conference on Learning Representations (ICLR)*, 2023.

Jiménez-Luna, J., Grisoni, F., and Schneider, G. Drug discovery with explainable artificial intelligence. *Nature Machine Intelligence*, 2(10):573–584, 2020.

Jumper, J., Evans, R., Pritzel, A., Green, T., Figurnov, M., Ronneberger, O., Tunyasuvunakool, K., Bates, R., Žídek, A., Potapenko, A., et al. Highly accurate protein structure prediction with alphafold. *Nature*, 596(7873):583–589, 2021.

Karniadakis, G. E., Kevrekidis, I. G., Lu, L., Perdikaris, P., Wang, S., and Yang, L. Physics-informed machine learning. *Nature Reviews Physics*, 3(6):422–440, 2021.

Katipoglu-Yazan, T. et al. A high-throughput dataset of E. coli NCM3722 growth across 18 temperature levels. *Data in Brief*, 2023.

Khanal, O., Helleckes, L. M., et al. Applications of machine learning in biopharmaceutical process development and manufacturing: Current trends, challenges, and opportunities. *arXiv preprint arXiv:2310.09991*, 2024.

Khuat, T. T. et al. Online machine learning for industrial cell culture monitoring. *Biotechnology and Bioengineering*, 2025.

Kusters, M. et al. Metabolic constraint pinn for e. coli biomass prediction. *NeurIPS Workshop on AI for Science*, 2025.

Li, X. et al. Process control for industrial fermentation: a review of recent advances. *Biochemical Engineering Journal*, 2024.

Lin, Z., Akin, H., Rao, R., Hie, B., Zhu, Z., Lu, W., Smetanin, N., Verkuil, R., Kabeli, O., Shmueli, Y., et al. Evolutionary-scale prediction of atomic-level protein structure. *Science*, 379(6637):1123–1130, 2023.

Luedeking, R. and Piret, E. L. A kinetic study of the lactic acid fermentation. batch process at controlled ph. *Journal of Biochemical and Microbiological Technology and Engineering*, 1(4):393–412, 1959.

Monod, J. The growth of bacterial cultures. *Annual Review of Microbiology*, 3:371–394, 1949.

Mowbray, M., Savage, T. R., Wu, C., Song, Z., Cho, B. A., del Rio-Chanona, E. A., and Zhang, D. Machine learning for biochemical engineering: A review. *Biochemical Engineering Journal*, 172:108054, 2021.

Peng, S.-T. et al. Systematic comparison of machine learning methods for bioprocess monitoring. *Biotechnology and Bioengineering*, 2025.

Petsagkourakis, P. et al. Reinforcement learning for batch bioprocess optimization. *Computers & Chemical Engineering*, 2020.

Pinto, J. et al. A general deep hybrid model for bioreactor systems. *Biochemical Engineering Journal*, 183:108440, 2022.

Raissi, M., Perdikaris, P., and Karniadakis, G. E. Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. *Journal of Computational Physics*, 378:686–707, 2019.

Ramos, J. et al. Hybrid lstm model for hek293 fed-batch. *Biotechnology and Bioengineering*, 2024.

Richter, F. et al. Predicting monoclonal antibody titer using machine learning. *Biotechnology and Bioengineering*, 2025.

Riezzo, R. et al. Hybrid neural–mechanistic models for CHO cell culture parameter estimation. *Biotechnology and Bioengineering*, 2025.

Ross, T. Indices for performance evaluation of predictive models in food microbiology. *Journal of Applied Bacteriology*, 81(5):501–508, 1996.

Rosso, L., Lobry, J., Bajard, S., and Flandrois, J. Convenient model to describe the combined effects of temperature and ph on microbial growth. *Applied and Environmental Microbiology*, 61(2):610–616, 1995.

Shah, P. et al. Deep neural network estimation of monod kinetics for industrial fermentation. *Biochemical Engineering Journal*, 2022.

Smiatek, J. et al. A snapshot of biomanufacturing and the need for enabling research infrastructure. *Trends in Biotechnology*, 2024.

Stokes, J. M., Yang, K., Swanson, K., Jin, W., Cubillos-Ruiz, A., Donghia, N. M., MacNair, C. R., French, S., Carfrae, L. A., Bloom-Ackermann, Z., et al. A deep learning approach to antibiotic discovery. *Cell*, 180(4): 688–702, 2020.

University College London Research Data Repository. BL21 (UCL): a pH $\times$ temperature $\times$ buffer batch growth dataset of *e. coli bl21*. UCL RDR `24039384`, 2024.

Vamathevan, J., Clark, D., Czodrowski, P., et al. Applications of machine learning in drug discovery and development. *Nature Reviews Drug Discovery*, 18(6):463–477, 2019.

Watson, J. L., Juergens, D., Bennett, N. R., Trippe, B. L., Yim, J., Eisenach, H. E., Ahern, W., Borst, A. J., Ragotte, R. J., Milles, L. F., et al. De novo design of protein structure and function with RFdiffusion. *Nature*, 620: 1089–1100, 2023.

Zaika, L. L. et al. Modeling the growth of *shigella flexneri* in food. *Journal of Food Protection*, 1994.

Zhu, Y. et al. Physics-informed neural networks for microbial growth modeling. *Bioresource Technology*, 2026.

Zwietering, M., Jongenburger, I., Rombouts, F., and Van't Riet, K. Modeling of the bacterial growth curve. *Applied and Environmental Microbiology*, 56(6):1875–1881, 1990.

A. Dataset Details

A.1. Naming convention

We adopt paper-facing dataset names throughout the paper, mapping each internal dataset identifier to the original source. Body text uses the full name on first mention with an organism qualifier (e.g., *Katipoglu-Yazan 2023* (*E. coli* NCM3722)) and the full name thereafter. Tables and figures may use the short form when space requires it; the short form is then defined in the caption.

A.2. Dataset statistics

Table 4 summarizes the 11 datasets used in the paper. *N raw* is the original curve count from the source repository; *N preproc* is the remaining curve count after the per-curve `with_growth` filter, which retains a curve only if its single-curve Monod-Baranyi nonlinear least-squares fit achieves $R^2 \geq 0.4$. Time horizon and time interval are means over the preprocessed set; the env axis lists the within-dataset varying conditions. *Avg* $\mu_{\max}$ is the per-curve fitted growth rate averaged across preprocessed curves and is used only as a coarse cross-organism comparator; the model never sees these literature fits during training.

Three datasets on which no baseline could reach a best mean $R^2 \geq 0.42$ at size L (Yeast Y1000+, *S. aureus* Buchanan, *S. aureus* ComBase) are excluded at the dataset level. *BL21 (UCL)* reaches a mid-tier best $R^2 \approx 0.346$ but is retained for its own-data value and for organism diversity.

A.3. Train, validation, and test split

We use curve-level random splits in which Test (15%) and Val (15%) are size-invariant and only Train varies across three size variants S/M/L (20% / 40% / 70% of total preprocessed curves). Splits are seeded by `(organism_id, seed_idx)` for `seed_idx` $\in \{0, 1, 2, 3, 4\}$ so that the five-seed average is computed over genuinely distinct partitions. Within Test, we additionally hold out the back 70% of timepoints per curve as the never-seen-as-context evaluation set (Section 4.3 in the main paper); all context densities therefore evaluate on identical timepoints. The paper's main results use size L only, so the training proportion is fixed at 70% and the full sweep is $11 \times 9 \times 5 = 495$ cells per `(sim_mode, ctx_ratio)` combination.

B. Simulation Generation Protocol

B.1. Simulation variants

This section defines the simulation curve synthesis protocol used in the RQ2 ablation. Following the three factors introduced in Section 3.3 of the main paper, each variant is fully specified by (Factor A) the ODE family that is integrated, (Factor B) the parameter distribution that is sampled, and (Factor C) the resulting position on the biokinetic specificity scale (random-GP control / single-ODE narrow / composite-ODE broad).

ODE families. We consider four biokinetic ODE families and one non-biokinetic control. Throughout, $N(t)$ is the cell density, $S(t)$ the substrate concentration, and $q(t)$ a lag-phase adaptation state.

Monod (Monod, 1949) models substrate-limited growth as

$$\begin{aligned} \frac{dN}{dt} &= \mu(S)\, N, \\ \mu(S) &= \mu_{\max} \tfrac{S}{K_s+S}, \\ \tfrac{dS}{dt} &= -\tfrac{\mu(S)\, N}{Y_{xs}}. \end{aligned} \tag{10}$$

Logistic ignores substrate dynamics and uses a carrying-capacity term:

$$\frac{dN}{dt} = \mu_{\max} N\big(1 - \tfrac{N}{K}\big)\,. \tag{11}$$

Gompertz (Zwietering et al., 1990) is a sigmoidal closed form

$$N(t) = N_0 + A\, e^{-\exp\left(\frac{\mu_{\max} e}{A}(\lambda - t) + 1\right)}, \tag{12}$$

with $A = K - N_0$, lag time λ, and $e = \exp(1)$.

Baranyi & Roberts (Baranyi & Roberts, 1994) introduces a two-state ODE with explicit lag dynamics:

$$\begin{aligned} \frac{dN}{dt} &= \mu_{\max} \frac{q}{1+q} \big(1 - \tfrac{N}{K}\big) N, \\ \frac{dq}{dt} &= \mu_{\max}\, q. \end{aligned} \tag{13}$$

Rosso composite (Rosso et al., 1995) couples the Monod substrate term with cardinal envelopes for temperature and pH:

$$\mu(S, T, \mathrm{pH}) = \gamma(T)\, \gamma(\mathrm{pH})\, \mu_{\max} \tfrac{S}{K_s+S}, \tag{14}$$

where each γ is the Rosso cardinal function with three reference points (e.g., $T_{\min}, T_{\mathrm{opt}}, T_{\max}$) and is positive only inside the biological envelope.

Parameter sampling. Each simulated curve draws parameters from a distribution centered at the organism's literature warm-start mean (the same values used for the BioStruct-ODE ParamHead bias initialization, Appendix C) $\pm$ a sampling spread. The *broad uniform* regime uses literature $\pm 50\%$, while the *literature-tight* regime uses literature $\pm 10\%$. The initial condition N_0 is sampled from $[0.01, 0.05]$ in normalized units, and we add a multiplicative log-normal observation noise ($\sigma = 0.05$).

Table 3. Dataset naming convention. Body text uses the full name on first mention with an organism qualifier, and the full name thereafter. Tables and figures may use the short form when space requires; the short form is then defined in the caption.

Paper full name	Short form	Source
BL21 (UCL)	BL21	UCL RDR 24039384
Katipoglu-Yazan 2023	Katipoglu23	*Data in Brief* 2023
Faure 2023	Faure23	Faure et al., 2023
Buchanan 1989	Buchan89	Buchanan et al., 1989 (ComBase)
Zaika 1994	Zaika94	Zaika et al., 1994 (ComBase)
Eifert 1997	Eifert97	Eifert et al., 1997 (ComBase)
ComBase (E. coli)	CB-Ec	ComBase src9788
ComBase (Listeria)	CB-Lm	ComBase src9788
ComBase (Yersinia)	CB-Yer	ComBase src9788
ComBase (Salmonella)	CB-Sal	ComBase src9788
ComBase (Pseudomonads)	CB-Pse	ComBase src6878

Table 4. Dataset statistics for the 11 datasets used in the paper. *N raw* = original curve count from the source repository; *N preproc* = remaining curves after the per-curve `with_growth` filter (single-curve Monod-Baranyi NLS fit $R^2 \geq 0.4$ retained). Time horizon and time interval are means over the preprocessed set; the env axis lists within-dataset varying conditions.

Dataset	Organism	N raw	N preproc	Horizon (h)	Interval (min)	Env axis
BL21 (UCL)	E. coli BL21	78	72	24 ± 6	5	pH × T × buffer
Katipoglu-Yazan 2023	E. coli NCM3722	504	480	18 ± 4	10.5	T (18 levels)
Faure 2023	E. coli DH5α	1024	920	18 ± 2	10	medium (10D binary)
Buchanan 1989	L. monocytogenes	508	470	200 ± 50	60	T × pH
Zaika 1994	S. flexneri	312	285	80 ± 20	30	T × pH
Eifert 1997	S. aureus	256	230	120 ± 30	30	T × NaCl
ComBase (E. coli)	E. coli	410	380	100 ± 25	30	T × pH
ComBase (Listeria)	Listeria spp.	380	350	180 ± 40	60	T
ComBase (Yersinia)	Yersinia	290	265	150 ± 35	60	T
ComBase (Salmonella)	Salmonella	320	295	90 ± 25	30	T × pH
ComBase (Pseudomonads)	Pseudomonads	240	220	110 ± 30	60	T

Table 5. Per-dataset train / validation / test split. Test (15%) and Val (15%) are size-invariant; the training set varies across three size variants S/M/L (20% / 40% / 70% of total preprocessed curves). Splits are at the curve level with random sampling, seeded by `(organism_id, seed_idx)` for `seed_idx` $\in \{0, 1, 2, 3, 4\}$. The paper's main results use size L only.

Dataset	Total	Train (S)	Train (M)	Train (L)	Val	Test
BL21 (UCL)	72	14	28	50	10	12
Katipoglu-Yazan 2023	480	96	192	336	72	72
Faure 2023	920	184	368	644	138	138
Buchanan 1989	470	94	188	329	70	70
Zaika 1994	285	57	114	200	43	43
Eifert 1997	230	46	92	161	35	35
ComBase (E. coli)	380	76	152	266	57	57
ComBase (Listeria)	350	70	140	245	53	53
ComBase (Yersinia)	265	53	106	185	40	40
ComBase (Salmonella)	295	59	118	207	44	44
ComBase (Pseudomonads)	220	44	88	154	33	33

Random-GP control. The *sim_random_gp* variant uses no biokinetic ODE at all. It draws length-scales $\ell \sim \mathrm{Uniform}[2, 12]$ h and samples from a zero-mean Gaussian process with an RBF kernel of scale ℓ, with no monotonicity, saturation, or other biological structure imposed. This control isolates whether the benefit of simulation pre-training is due to *biokinetic specificity* per se or merely to exposing the model to many smooth curves.

Variant table. The four sim variants used in our experiments are summarized in Table 6.

C. Baseline Architecture and Implementation

C.1. Architecture and parameter counts

Table 7 lists the encoder, decoder, hidden / latent dimensions, and parameter counts for all nine baselines at three size vari-

Table 6. Simulation dataset variants used in RQ2. Per-organism sampling centers each parameter at the literature warm-start mean ± the indicated sampling spread; *sim_random_gp* is a non-biokinetic control (Gaussian Process samples). The fourth row, *sim_per_family*, is used for the RQ2 ODE-family ablation with one ODE family per variant.

Sim variant	ODE family	Sampling	N curves
`sim_composite_broad`	Monod + Baranyi + Gompertz mixture	broad uniform (literature $\pm 50\%$)	5000
`sim_single_narrow`	Monod-Baranyi only	literature $\pm 10\%$	5000
`sim_random_gp`	none — Gaussian Process	random GP, length-scale $\in [2, 12]$ h	5000
`sim_per_family`	one of {Monod, Logistic, Gompertz, Baranyi, Rosso}	broad	5000 each

Parameter ranges (sim_composite_broad / sim_single_narrow): $\mu_{\max} \in [0.05, 1.50]/[0.40, 0.95]\ h^{-1}$; $K \in [0.10, 1.50]/[0.50, 1.00]$ *(normalized); lag* $\in [0, 8]/[0, 4]$ *h.*

ants. All neural baselines share the same encoder stack (ENVENCODER for environmental conditions and CONTEXTENCODER for partial-observation context) and differ only in the decoder head, which isolates the architectural-prior contribution from encoder capacity. The *PINN* baseline reuses the MLP decoder verbatim and adds an autograd-based physics-residual auxiliary loss term during training (defined in Appendix D.1). The *ODE-Guide*, *Hybrid-NeuralODE*, and BioStruct-ODE family progressively integrate a biokinetic ODE into the forward pass with increasing structure (closed-form template → two-state ODE with neural correction → three-state Monod-Baranyi with ParamHead and Rosso cardinal env-conditioning). The *Mean* and *ODE-Fit* baselines are non-parametric and have no learned weights.

The *ODE-Fit* baseline fits a Gompertz ODE per training curve with nonlinear least squares; the per-curve parameters (μ, K, λ, N_0) are averaged across the dataset's training set, and the averaged parameters are applied to every test curve of the same dataset. This protocol uses no environmental conditioning, paralleling the *Mean* baseline but with biokinetic structure (Appendix D.1 defines the L_4 Huber regression loss used by all neural baselines). The BioStruct-ODE family PARAMHEAD biases are warm-started from organism-specific literature Monod fits.

D. Training Details

D.1. Loss functions

All neural baselines are trained against a regression loss on the observed timepoints. We consider five loss families and adopt L_4 (Huber) as the paper-wide default based on a preliminary loss ablation; let $\Omega = \{(i,t)\}$ denote the set of (curve, timepoint) observations. The MSE and MAE losses are

$$\mathcal{L}_{L0} \;=\; \tfrac{1}{|\Omega|} \sum_{(i,t)\in\Omega} \big(\hat{y}_i(t) - y_i(t)\big)^2, \tag{15}$$

$$\mathcal{L}_{L1} \;=\; \tfrac{1}{|\Omega|} \sum_{(i,t)\in\Omega} \big|\hat{y}_i(t) - y_i(t)\big|. \tag{16}$$

The time-consistency and curvature regularizers add a derivative penalty to the MSE base,

$$\mathcal{L}_{L2} \;=\; \mathcal{L}_{L0} + \lambda_{\text{tc}} \sum_{(i,t)} \big|\partial_t \hat{y}_i(t)\big|^2, \tag{17}$$

$$\mathcal{L}_{L3} \;=\; \mathcal{L}_{L0} + \lambda_{\text{c}} \sum_{(i,t)} \big|\partial_t^2 \hat{y}_i(t)\big|^2. \tag{18}$$

The default Huber loss with threshold $\delta = 0.1$ is

$$\mathcal{L}_{L4} \;=\; \tfrac{1}{|\Omega|} \sum_{(i,t)\in\Omega} \rho_\delta\big(\hat{y}_i(t) - y_i(t)\big). \tag{19}$$

The *PINN* baseline adds an auxiliary physics-residual loss in the form of a logistic ODE,

$$\mathcal{L}_{\text{phys}} \;=\; \frac{1}{|T_c|} \sum_{t\in T_c} \left(\frac{d\hat{y}}{dt} \;-\; \mu\, \hat{y}\,(1 - \hat{y}/K) \right)^2, \tag{20}$$

where (μ, K) are taken from the per-dataset Gompertz fits used by the *ODE-Fit* baseline (so the loss-channel and template-channel see *the same* kinetic prior, isolating the channel comparison), T_c is a random sample of collocation timepoints drawn each minibatch, and $d\hat{y}/dt$ is computed by automatic differentiation. The total PINN training loss is $\mathcal{L}_{\text{total}} = (1 - \lambda)\,\mathcal{L}_{\text{data}} + \lambda\,\mathcal{L}_{\text{phys}}$ with $\lambda = 0.1$ (no per-dataset λ tuning).

D.2. Optimizer, schedule, and computational budget

We use AdamW with learning rate 3×10^{-4}, weight decay 10^{-5}, linear warmup over the first 10 epochs, and cosine decay thereafter. Training is monitored on validation R^2 at `ctx 0%` with early stopping (patience 5, $\Delta_{\min} = 10^{-4}$). All experiments run on 8× NVIDIA RTX 3090 GPUs at five concurrent workers per GPU (40 workers total); the full RQ1 sweep (9 baselines × 11 datasets × 5 seeds × size L) completes in approximately one wall-clock hour. Aggregate compute budgets are RQ1 ≈ 16 GPU-hours, RQ2 ≈ 80 GPU-hours, RQ3 ≈ 24 GPU-hours, and RQ4 (post-hoc analysis only) ≈ 0, for a total of about 120 GPU-hours of paper-supporting compute.

Table 7. Per-baseline architecture and parameter counts at three size variants (S / M / L). All neural baselines share the same encoder stack (EnvEncoder + ContextEncoder) and differ only in the decoder head; this isolates the architectural-prior contribution from encoder capacity. Mean and ODE-Fit are non-parametric baselines without learned weights.

Model	Encoder	Decoder	Hidden	Params (S/M/L)
Mean	—	mean trajectory	—	0 / 0 / 0
ODE-Fit	—	NLS Monod–Baranyi	—	4 / 4 / 4
MLP	shared	4-layer MLP	32/64/96	12K / 48K / 180K
GRU	shared	2-layer GRU	32/64/96	25K / 90K / 320K
PINN	shared	MLP + physics loss	32/64/96	12K / 48K / 180K
ODE-Guide	shared	Gompertz + MLP residual	32/64/96	18K / 65K / 240K
Hybrid-NeuralODE	shared	Baranyi ODE + correction	32/64/96	22K / 80K / 290K
BioStruct-ODE (Ours)	shared	3-state ODE + ParamHead	32/64/96	24K / 92K / 340K
BioStruct-ODE+Cardinal (Ours)	shared	+ Rosso envelope	32/64/96	26K / 100K / 360K

shared = EnvEncoder + ContextEncoder; Hidden is the per-size hidden dim.

E. Ablation Studies

E.1. Biokinetic component ablation

Table 8 reports the contribution of each binary biokinetic component (warm-start, death term, template choice) on the two main bio-structured models. Each component is toggled while the other two are held at the paper default (Warm-start on, Death term on, Template = Monod-Baranyi); the default row matches Table 1 exactly for both models.

The death term contributes the largest single Af improvement (1.330 $\rightarrow$ 1.266 for BioStruct-ODE; 1.290 $\rightarrow$ 1.235 for BioStruct-ODE+Cardinal), with a smaller R^2 lift of $+0.011$ to $+0.012$; its primary effect is on stationary-phase fidelity. Warm-start gives the smallest R^2 gain ($+0.006$ to $+0.008$) but a consistent A_f improvement of about -0.02 to -0.03, stabilizing the kinetic head without dramatic effect. Monod-Baranyi outperforms Gompertz by a stable margin ($\approx +0.02$ R^2, ≈ -0.03 A_f) across both models, since the substrate-dynamics term in Monod-Baranyi provides a richer prior than the logistic-only Gompertz form.

E.2. Cross-organism transfer

We evaluate cross-organism transfer with BioStruct-ODE (Ours, scratch) by training on each organism's training set and evaluating on every other organism's test set, producing the 11×11 matrix in Table 9. The bold diagonal corresponds to the within-organism baseline and matches the per-dataset BioStruct-ODE means in Table 1.

The off-diagonal mean (0.108) is far below the diagonal mean (0.572), giving a gap of $\Delta = -0.464$. Phylogenetically close organisms recover only 30–40% of the within-organism R^2 (e.g., the four *E. coli* variants among themselves), and transfer to non-*E. coli* organisms is essentially uniform near $R^2 < 0.2$. This is the direct quantitative basis for adopting a per-organism scope in the main paper.

Table 8. Biokinetic component ablation on the two bio-structured models (Ours). Each component is toggled binary while keeping the other two at the paper default (Warm-start on, Death term on, Template = Monod–Baranyi); the default row matches Table 1 exactly. All three components contribute small but consistent gains, and the death term yields the largest single A_f improvement.

		$R^2 \uparrow$		$A_f \downarrow$	
Component	**State**	**BioStruct-ODE**	**+Cardinal**	**BioStruct-ODE**	**+Cardinal**
Warm-start	on (default)	**0.521**	**0.554**	**1.266**	**1.235**
	off	0.515	0.546	1.290	1.255
Death term	on (default)	**0.521**	**0.554**	**1.266**	**1.235**
	off	0.510	0.542	1.330	1.290
Template	Monod-Baranyi (default)	**0.521**	**0.554**	**1.266**	**1.235**
	Gompertz	0.500	0.532	1.295	1.265

Table 9. Cross-organism transfer matrix for BioStruct-ODE (Ours, scratch). Cell value $= R^2$ when training on Source organism's training set and evaluating on Target organism's test set. **Bold diagonal** = within-organism baseline (matches the per-dataset BioStruct-ODE values in `t1_per_cell.csv`). Off-diagonal mean (0.108) is far below the diagonal mean (0.572), giving a $\Delta = -0.46$ that justifies the per-organism scope of the main paper. Short forms: BL21 = BL21 (UCL); Katipoglu23 = Katipoglu-Yazan 2023; Faure23 = Faure 2023; Buchan89 = Buchanan 1989; Zaika94 = Zaika 1994; Eifert97 = Eifert 1997; CB-Ec/Lm/Yer/Sal/Pse = ComBase (E. coli / Listeria / Yersinia / Salmonella / Pseudomonads).

Source ↓ \ Target →	**BL21**	**Katipoglu23**	**Faure23**	**Buchan89**	**Zaika94**	**Eifert97**	**CB-Ec**	**CB-Lm**	**CB-Yer**	**CB-Sal**	**CB-Pse**
BL21	**.29**	.18	.20	.05	.10	.05	.21	.06	.09	.16	.05
Katipoglu23	.20	**.83**	.22	.06	.11	.05	.24	.07	.10	.18	.06
Faure23	.19	.21	**.88**	.05	.09	.04	.20	.06	.08	.14	.04
Buchan89	.06	.07	.05	**.60**	.12	.08	.07	.18	.10	.09	.05
Zaika94	.11	.10	.09	.13	**.42**	.06	.10	.10	.12	.11	.05
Eifert97	.05	.06	.04	.09	.07	**.62**	.05	.08	.07	.06	.04
CB-Ec	.22	.24	.21	.06	.10	.05	**.55**	.07	.09	.18	.05
CB-Lm	.06	.07	.05	.18	.10	.08	.07	**.48**	.11	.09	.05
CB-Yer	.09	.10	.08	.10	.11	.07	.09	.10	**.42**	.10	.05
CB-Sal	.17	.18	.15	.09	.11	.06	.18	.09	.10	**.56**	.05
CB-Pse	.06	.06	.05	.05	.05	.04	.05	.05	.05	.05	**.65**