PROGRESSIVE CONDITIONAL DENOISING AUTOENCODER AS A PHYSICAL SURROGATE MODEL

José Afonso [*], **Vasco Guerra**[1], and **Pedro Viegas**[1]

[1]Instituto de Plasmas e Fusão Nuclear, Instituto Superior Técnico, Av. Rovisco Pais 1, Lisbon, Portugal

ABSTRACT

Surrogate models developed using Physics-Informed Neural Networks (PINNs) can present difficulties in training and do not explicitly learn the geometric structure of the physical manifold where solutions reside. To address this limitation, we introduce the Progressive Conditional Denoising Autoencoder (PCDAE), a generative model that circumvents these issues by implicitly learning the manifold of valid physical solutions. The PCDAE is trained to reconstruct clean data from noisy inputs across a range of signal-to-noise ratios, effectively learning a vector field that guides solutions back to the data manifold. On a plasma physics benchmark, the PCDAE demonstrates substantially higher parameter and data efficiency than a baseline PINN with the Projection method. We find that our model's implicit grasp of the system's physics is complete enough so that enforcing explicitly physical conservation laws during inference provides no additional benefit. Our implementation is available at https://github.com/joseAf28/PCDAE-SurrogatePhysicalModel.

***K*eywords** Scientific Machine Learning · Surrogate Modeling · Generative Models

1 Introduction

Generative models are a class of deep learning algorithms designed to estimate and sample from an unknown data distribution. The remarkable advances in this field, particularly for image generation, have been largely driven by the scalable and stable training of diffusion-based and auto-encoding models [1, 2, 3]. These models excel at capturing the global, intrinsic structure of a data manifold without imposing rigid factorizations, unlike their autoregressive counterparts [4]. However, the success of these score-based methods is fundamentally reliant on massive datasets, making them challenging to apply in specialized scientific domains where data is often scarce and expensive to generate.

In scientific domains like Low-Temperature Plasma Physics, simulations are critical but computationally demanding. Simulators such as LoKI [5] model plasma physics phenomena by solving complex systems of differential equations. For many applications, the steady-state solution is of primary interest, which requires finding the fixed y of a system of algebraic equations:

$$F(x, y; \theta) = 0 \tag{1}$$

Here, F represents the system of equations derived from the kinetic scheme [6], x represents the experimental input conditions, y the resulting chemical densities, and θ the physical hyperparameters of the simulator. The computational cost of finding these solutions for numerous conditions is often intractable, making the development of fast and accurate surrogate models a necessity.

A straightforward approach to creating a surrogate is to train a direct mapping $f(x) \approx y$ using a standard neural network. However, such models are trained as *black-box* regressors and are inherently physics-agnostic. Their predictions are mere point estimates that lack any guarantee of satisfying the fundamental physical laws embedded in the function F, potentially leading to physically meaningless results.

To address this, the paradigm of Physics-Informed Machine Learning has emerged [7]. A popular approach within this paradigm is the Physics-Informed Neural Network (PINN), which incorporates physical constraints directly into

[*]Corresponding author: josefafonso@tecnico.ulisboa.pt

the loss function as a soft penalty term [7, 8]. The goal is to guide the model's training towards physically consistent solutions. However, this method often suffers from significant training pathologies. The gradients from the data-fitting term and the physics-based penalty term can conflict, leading to training instabilities, slow convergence, or a failure to learn the underlying relationships accurately [9]. The physical constraints, rather than aiding the learning process, can hinder it or have no impact.

An alternative strategy is the projection method, where a model is trained conventionally, and its output $\hat{y}$ is then projected onto the constraint manifold in a post-processing step [8]. This is achieved by solving an optimization problem that finds the closest physically valid point p to the model's prediction. While this ensures the final output is always physically consistent, it decouples the physical knowledge from the learning process itself. The model does not inherently learn the structure of the physical manifold; it relies on the projection as a corrective crutch. A further limitation is that the constraint manifold is often defined by broad, global conservation laws (e.g., mass or energy conservation), which can be too generic to enforce the detailed, local physics of the system.

This work explores a different paradigm: learning the physical manifold implicitly. Instead of explicitly enforcing physical laws through loss terms or post-hoc projections, we propose a surrogate model based on a Progressive Conditional Denoising Autoencoder (PCDAE) [10]. This approach avoids the gradient conflicts seen in PINNs while integrating extra data-driven physical consistency directly into the learning process.

2 Background

The core premise of our approach is that for a given experimental conditions x, the corresponding valid physical solutions y do not occupy the entire output space. Instead, they are constrained to a lower-dimensional, highly structured subspace known as the data manifold [3, 10, 11]. Any point lying off this manifold represents a physically inconsistent or impossible state. The central challenge for a surrogate model is to implicitly learn the underlying shape and geometry of the physical manifold.

A Conditional Denoising Autoencoder (CDAE) is a smart framework for achieving this. Rather than learning a direct mapping from x to y, a CDAE learns the underlying geometry of the physical manifold itself. The training procedure is intuitive: it starts by taking a valid data point y that lies on the manifold and intentionally corrupting it with Gaussian noise, $y_\sigma = y + \sigma\epsilon$, where $\epsilon \sim \mathcal{N}(0, I)$. This perturbation pushes the point off the manifold into the surrounding, higher-dimensional space [10, 11, 12].

The task of the neural network, g_ϕ, is to take this corrupted, off-manifold point y_σ and the guiding condition x as input, and learn to project it back to its original, physically valid location y on the manifold. In this way, the model g_ϕ is forced to implicitly learn the intrinsic structure, constraints, and correlations that define the manifold's geometry [11]. It learns a vector field that effectively pulls any point in the vicinity of the manifold back onto it.

This learning process is driven by minimizing the mean squared error (MSE) between the network's prediction and the original clean data:

$$\mathcal{L}(\phi) = \mathbb{E}_{x,y,\epsilon}\left[\|g_\phi(y + \sigma\epsilon, x) - y\|_2^2\right]. \tag{2}$$

3 Progressive Conditional Denoising Autoencoder (PCDAE)

3.1 Suboptimality of a Fixed Noise Level σ

However, in practice, training a CDAE with a single noise level σ is suboptimal for learning the vector field that reconstructs the data manifold. On the one hand, if the noise is too small, the model only learns to make minor corrections. It becomes an expert at local details but fails to learn the global structure of the data distribution [3]. On the other hand, if the noise is too large, fine-grained details of the data are washed out. The model learns the coarse, global structure but produces blurry or imprecise reconstructions that lack high-fidelity details [10, 11].

To overcome this, we train the CDAE across a continuum of noise levels, allowing the model to learn both the global manifold structure (at high σ) and fine details (at low σ) as widely employed in diffusion models [2, 3].

As a result, we obtain the following objective loss:

$$\mathcal{L}(\phi) = \mathbb{E}_{(x,y)\sim p_{\text{data}}}\mathbb{E}_{\sigma\sim p_\sigma}\mathbb{E}_{y_\sigma\sim q_\sigma(\cdot|y)}\left[\|g_\phi(y_\sigma, x, \sigma) - y\|_2^2\right], \tag{3}$$

where $q_\sigma(\cdot \mid y) = \mathcal{N}(y, \sigma^2 I)$ and the noise level σ is drawn from $p_\sigma(\cdot)$,

Algorithm 1 summarizes the training procedure we adopt for training the PCDAE.

3.2 Theoretical Guarantees

The Mean Squared Error (MSE) objective used to train our denoising model (Eq. [2]) is well-founded in probabilistic theory. This objective is equivalent to maximizing the *recovery log-likelihood* [11, 13] under a Gaussian posterior assumption, grounding our approach in the principle of maximum likelihood estimation.

Furthermore, this objective provides an effective implementation of denoising score matching. In the limit of infinitesimal noise ($\sigma \to 0$), the score of the noisy data distribution, which the model implicitly learns, converges to the score of the true data distribution $\nabla_Y \log p(Y|X)$ [11]. Thus, our simple objective provides the dual benefits of maximum likelihood and score-based modeling, ensuring the model learns a robust representation of the true data manifold.

3.3 Comparison with Flow Matching

While the Denoising and Flow Matching [14] frameworks are deeply related ways to model the transformation from noise to data, their subtle differences in objectives lead to significant practical consequences for surrogate modeling.

The denoising objective (Eq.[2]) is used to train the model g_ϕ that learns the mapping from an experimental condition X to its physical output Y_1 (we redefine $Y_1 = Y$, $Y_0 = \epsilon \sim \mathcal{N}(0, I)$ and $\sigma_t = t'$ for convenience). Given a noisy sample $Y_{t'}$ and the condition X, the model learns to approximate the conditional expectation of the clean data [11]:

$$g^*_\phi(Y_t, X, t') = \mathbb{E}[Y_1 \mid Y_{t'}, X] \tag{4}$$

where $Y_{t'} = Y_1 + t' \, Y_0$ which corresponds to the *variance exploding* scheme [15]. This means the model's output is an estimate of the true output Y_1. Then, the vector field implicitly defined by the CDAE constitutes an approximation to:

$$v(Y_{t'}, X, t') = \mathbb{E}[Y_1 \mid Y_{t'}, X] - Y_{t'} \tag{5}$$

as widely explored in [11]. The generative process uses this vector field to perform a series of update steps. Starting from y^τ, we can obtain a more refined estimate $y^{\tau+1}$ by taking a small step in the direction of the vector field with the noise level σ_k:

$$y^{\tau+1} = y^\tau + \eta \, v(y^\tau, x, \sigma_k). \tag{6}$$

Here, η stands for a hyperparameter that controls the step size during our refinement process. In this way, the corrective vector field defines a trajectory that guides the samples to the data manifold, always conditioned on x. Furthermore, we implement annealed inference, as outlined in [3]. Algorithm 2 summarizes the inference procedure used for generating predictions.

In contrast, the Flow Matching framework [14] trains a model v_ϕ to predict the vector field that transports noise to data distributions directly. For the optimal transport case, where we have a simple linear path, $Y_t = (1-t)Y_0 + t\,Y_1$ with $t \in [0, 1]$, the objective used is to approximate the conditional expectation of the transport path, v, [14]:

$$v(Y_t, X, t) = \mathbb{E}[Y_1 - Y_0 \mid Y_t, X] \tag{7}$$

where, the model's output is not the final state but rather the instantaneous direction of transport. Once the conditional vector field $v_\phi(y, x, t)$ is trained, new samples are generated by solving the corresponding probability flow ordinary differential equation (ODE) [14]:

$$\frac{dy_t}{dt} = v_\phi(y_t, x, t). \tag{8}$$

To generate a sample for a given condition x, an initial state y_0 is drawn from a standard normal distribution, $\mathcal{N}(0, I)$. The ODE is then numerically integrated from $t = 0$ to $t = 1$, and the resulting state, y_1, constitutes the final generated sample [14].

Our refinement procedure, outlined in Eq. [6], can be understood as the Euler discretization of an ODE, where the vector field is learned indirectly by the PCDAE model, rather than through a direct vector field regression as seen in Flow Matching, motivating the choices presented in Algorithm 2.

Moreover, for deterministic and low-data regimes, the choice of the PCDAE model over Flow Matching is justified by the statistical efficiency of its learning objective. Both methods train a model via regression, and for a finite dataset, the accuracy of the learned estimator is limited by the variance of the regression target [16, 17].

Given X, noisy samples Y_t and that we are dealing with deterministic datasets, it implies that the learning target for the PCDAE, $Y_1 \mid X$, has zero variance (delta-shaped). In contrast, the target for Flow Matching, $Y_1 - Y_0$, remains a stochastic variable due to the random noise term Y_0, resulting in a learning problem with a significantly higher target variance. As a result, the PCDAE's lower-variance objective provides a more statistically efficient and stable learning signal, making it a well-suited choice for the deterministic and low-data settings common in physical models.

3.4 Denosing as a Generalization of Regression

The denoising objective can be understood as a principled generalization of the standard regression task when viewed through the information-theoretic lens of the Information Bottleneck (IB) principle [18, 19]. A typical regression aims to learn the conditional expectation $\mathbb{E}[Y_1 \mid X]$, which represents the optimal prediction of Y_1 using only the information available in the condition X. Our framework learns the more general quantity $\mathbb{E}[Y_1|Y_{t'}, X]$, where the noise level t' acts as a parameter controlling an information bottleneck between the clean data Y_1 and its noisy version $Y_{t'}$.

In the high-noise limit, as $t' \to \infty$, the noisy sample $Y_{t'} = Y_1 + t'\, Y_0$ becomes saturated by noise, causing the mutual information $I(Y_1; Y_{t'})$ to vanish. Thus, $Y_{t'}$ offers no useful information for predicting Y_1, forcing the model to rely exclusively on X. In this regime, the learning objective collapses to the classical regression task: $\mathbb{E}[Y_1 \mid Y_{t'}, X] \approx \mathbb{E}[Y_1 \mid X]$ [18].

Conversely, in the zero-noise limit, $Y_{t'} = Y_1$ and the task of predicting Y_1 given Y_1 and X becomes a trivial mapping: $\mathbb{E}[Y_1 \mid Y_1, X] = Y_1$. Nonetheless, training in this regime is critical for anchoring the model to the data manifold and learning fine-grained details [12].

The true power of this framework is realized in the intermediate-noise regime, where $Y_{t'}$ provides partial, corrupted information about Y_1. As a result, the neural network learns to predict Y_1 by combining the global guidance from the condition X with the local consistency enforced by $Y_{t'}$, which provides information about the local geometry of the manifold.

Algorithm 1 PCDAE Training Algorithm

```
Input  : Data Dist $p_D(x, y)$ , Noise Distribution $p_\sigma(\cdot)$,
         Noise kernel $q_\sigma()$, PCDAE $g_\phi$
Output : Trained PCDAE $g_\phi$
while not converged do
    ▷ Generate samples:
    $x_i, y_i \sim p_D$
    $\sigma \sim p_\sigma$
    $\tilde{y}_i \sim q_\sigma(\cdot \mid y_i)$
    ▷ Optimize objective $\mathcal{L}$ wrt $\phi$:
    $\Delta\phi \leftarrow \nabla_\phi \sum_{i=1}^{N} \|g_\phi(\tilde{y}_i, x_i; \sigma_i) - y_i\|_2^2/N$;
    Update $\phi$ based on $\Delta\phi$ using Adam optimizer
end
```

Algorithm 2 PCDAE Inference Algorithm

```
Input  : Data Dist $p_D(x)$, PCDAE $g_\phi$, Size step $\eta$, Maximum Threshold $\epsilon_{\max}$,
         Number of Steps per level $K$, Noise Schedule $\{\sigma_i\}_{i=1}^{T}$ with
         $\sigma_1 > \sigma_2 > \cdots > \sigma_T$, Normal distribution $\mathcal{N}(0, I)$
▷ Sample input from $p_D$ and initialize candidate solution:
$x_i \sim p_D$, $\tilde{y}_i \sim \mathcal{N}(0, I)$
for each $\sigma_i$ $i = 1$ to $N$ do
    for run K steps of optimization $k = 1$ to $K$ do
        ▷ Compute the vector field $v$ update:
        $v \leftarrow g_\phi(\tilde{y}_i, x_i; \sigma_i) - \tilde{y}_i$
        ▷ Clip the vector field for stability:
        if $\|v\|_2 > \epsilon_{\max}$ then
            $v \leftarrow v \,.\, \epsilon_{\max}/\|v\|_2$
        end
        $\tilde{y}_i \leftarrow \tilde{y}_i + \eta\, v$
    end
end
return $\hat{y} \leftarrow \tilde{y}_N$
```

3.5 Physics-Constrained Inference

While the standard refinement algorithm learns the data manifold, the process is purely data-driven and does not guarantee strict adherence to known physical laws. To enforce physical consistency, we modify each inference step to explicitly incorporate these constraints (e.g., mass and energy conservation), $h(y) = 0$.

We achieve this by reformulating each step as a constrained optimization problem, which we solve using a primal-dual update on its Augmented Lagrangian, following the approach in [20]. This replaces the original update (Eq.[6]) with a new rule in Algorithm 2 that balances a *generative* step, guided by the learned model, with a *physical correction* step. It is given by:

$$y^{\tau+1} = y^{\tau} + \eta\,(g_\phi(y^{\tau}, x; \sigma) - y^{\tau}) - \alpha \nabla h(y^{\tau})^T(\lambda^{\tau} + \rho\, h(y^{\tau})), \tag{9}$$

$$\lambda^{\tau+1} = \lambda^{\tau} + \beta\, h(y^{\tau+1}), \tag{10}$$

where λ is the vector of Lagrange multipliers that dynamically enforces the constraints, while α and β are step-size hyperparameters.

The update for y now includes the physical correction, ensuring the solution is progressively guided towards both the learned data manifold and the explicit constraint manifold. The full derivation is detailed in the Appendix A.

4 Experiments

Here, we present an empirical evaluation of our proposed PCDAE model.

4.1 Experimental Setup

Dataset: We evaluate our model on a low-temperature plasma dataset introduced by [8], generated with the LisbOn Kinetics (LoKI) simulator [5]. This dataset contains 3000 samples, mapping 3 input experimental conditions (pressure, current, and the container radius) to 17 output chemical densities. As a pre-training procedure, all input and output features are normalized using Standard Scaling to have zero mean and unit variance.

PCDAE architecture: It consists of an encoder-decoder architecture. The encoder independently processes the three inputs: the condition x, the noisy data y_σ, and the noise level σ. The noise level is encoded using a transformer-style sinusoidal positional embedding to represent the continuous nature of σ [2, 21]. The resulting feature vectors are then concatenated. The decoder processes this unified representation, starting with layer normalization, to predict the clean data y concatenated with the input condition x. The full architectural details are provided in the Appendix B.

Baseline Model: We compare our model (which is physics-agnostic) against a Physics-Informed Neural Network (PINN) enhanced with the projection method [8]. To ensure a rigorous comparison, our PINN baseline replicates the architecture and physical loss functions described in [8]. Differing from our PCDAE model, this includes the explicit enforcement of mass and charge conservation laws, providing a strong and well-motivated baseline. Moreover, both models are trained and tested on identical datasets.

Evaluation Metrics: The performance of the model is assessed using the Root Mean Square Error (RMSE), which measures the difference between the predicted and actual chemical densities in the test set. To enhance the robustness of the results, we train an ensemble of models using 10 different train-test-validation splits. We report the mean results along with their standard deviation.

4.2 Results

We start by presenting the model efficiency comparison between the PCDAE and the PINN with Projection Baseline.

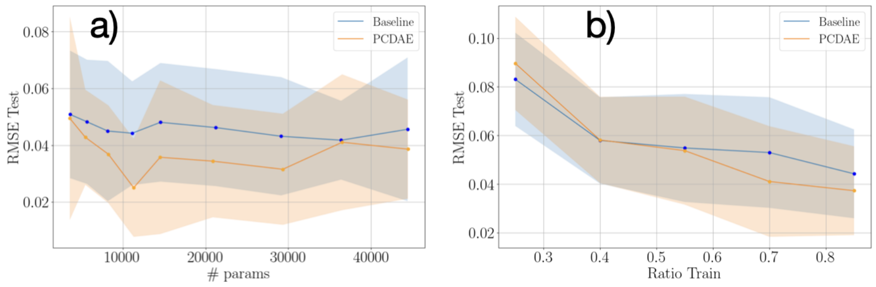

Figure 1: Efficiency comparison between our PCDAE (physics-agnostic, using Algorithm 1 and 1) (orange) and the PINN with Projection Baseline (blue). (a) Parameter efficiency, showing the test RMSE as a function of model size. (b) Data efficiency, showing the test RMSE as a function of the training data fraction for models of a fixed size. For panel (a), the PCDAE size was varied by widening the y encoder projection layer, while the Baseline's hidden layers were widened. Solid lines represent the mean performance over 10 independent runs, and shaded areas denote the standard deviation.

Fig. 1 a) demonstrates the superior parameter efficiency of our PCDAE model compared to the Baseline. The PCDAE consistently achieves a lower test RMSE than the Baseline across various model sizes, reaching optimal performance at around 12,000 parameters. This indicates that the PCDAE's architecture is more effective in capturing the physical data manifold's structure, allowing for higher accuracy with fewer parameters. While the mean performance is comparable for very small models (below 8,000 parameters), the PCDAE exhibits higher variance, suggesting a minimum capacity is needed for robust learning of the data manifold.

Fig.1 b) reveals an interesting trade-off. In the very low-data regime (below a 40% training ratio), the baseline model exhibits a slightly lower test error, indicating that the implicit task of learning the entire data manifold, as our PCDAE does, is inherently more complex and data-intensive than the simpler regression performed by the Baseline. With insufficient data to fully define the manifold's geometry, the generative model's performance is constrained. However, once this critical data threshold is surpassed (above 40%), the significant benefit of having learned the manifold becomes clear. The PCDAE's performance improves and surpasses the Baseline as more training data becomes available. This indicates that with a sufficient number of samples, the learned manifold acts as a powerful and accurate prior, leading to more robust and generalizable predictions, despite being physics-agnostic.

We perform the ablation study of the hyperparameters used during training and inference.

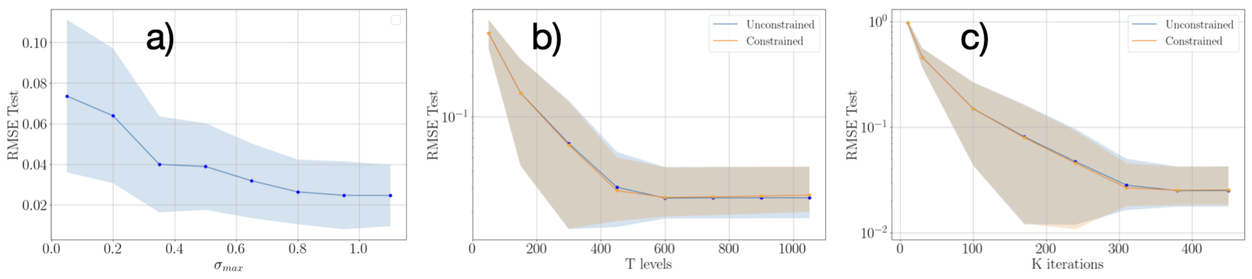

Figure 2: Ablation studies on PCDAE hyperparameters. (a) Test RMSE as a function of the maximum training and inference noise level, $\sigma_{\max}$. (b) Test RMSE as a function of the number of noise levels, T, for a fixed number of iterations per level $K = 10$ number of levels at inference time. (c) Test RMSE as a function K, for a fixed $T = 15$ used for the refinement procedure. Panels (b) and (c) compare the *unconstrained* (blue) and *constrained* (orange) inference methods. All experiments were conducted with a PCDAE model of $\sim 15,000$ parameters. Solid lines represent the mean performance over multiple runs, and shaded areas denote the standard deviation.

Fig.2 a) shows that both the mean and standard deviation of the test RMSE decrease as $\sigma_{\max}$ increases, up to the optimal point ($\sigma_{\max} \sim 0.9$). This finding strongly supports the manifold learning hypothesis [3, 10]. A larger $\sigma_{\max}$ forces the model to learn the global structure of the data manifold by training it to denoise highly corrupted samples that lie far from it. Conversely, in the case that the model is only exposed to small perturbations (a low $\sigma_{\max}$), it fails to learn the long-range corrective field necessary to guide the inference process from distant, noisy states, resulting in poorer and less stable predictions.

We also study the impact of the inference-time hyperparameters, the number of noise levels T, having fixed the number of iterations per level K, and the opposite case, we vary the number of refinement levels, K, while fixing the steps per level, T. As shown in Fig.2 b) and 2 c), the model's performance improves as K and T increase, eventually reaching a plateau where additional steps yield diminishing returns. The regime of Fig. 2 b) is more similar to the one used in diffusion models [2], while Fig. 2 c) is more related to the one used to sample from energy-based models [3, 13].

In this context, we compared our *unconstrained* inference (follows Eq. [6]) with the *constrained* one, which enforces explicit physical constraints (follows Eq. [10]) at inference time. Surprisingly, the primal-dual correction did not provide a significant benefit and, in fact, led to a slight degradation in performance. We found that the method was highly sensitive to its hyperparameters (α, β, ρ), with larger step sizes causing a catastrophic drop in accuracy.

A possible interpretation is that the explicit constraints enforced, which represent global conservation laws, are a generic and small subset of the complex local correlations the PCDAE has already learned implicitly from the data manifold [10, 11, 22]. The model has learned a highly specific corrective vector field to guide samples to the true, high-fidelity manifold. Imposing an additional, more general constraint perturbs this learned field, effectively pulling the solution towards a slightly different and less precise manifold, which degrades performance [10, 11]. This suggests that when a model has successfully learned the data distribution, simple post-hoc physical projections may be redundant or even harmful, especially when the model is not required to extrapolate far beyond the training distribution, where such physical constraints might otherwise provide a useful guardrail [20, 23].

To provide a deeper insight into the dynamics of the generative process, we analyzed the evolution of the absolute residuals for each of the 17 output dimensions during inference.

Fig.3 plots the mean (a) and standard deviation (b) of these residuals across the entire test set as a function of the denoising *time*, $\sigma_{\max} - \sigma(t)$, where t progresses from high noise to low noise.

As the refinement progresses, both the mean and the standard deviation of the absolute value of these residuals for all dimensions monotonically decrease by approximately two orders of magnitude. The sharp drop in the standard deviation, shown in panel (b), demonstrates a variance collapse. This phenomenon is a direct consequence of our deterministic training set, where each input condition x corresponds to a single ground-truth solution y. The model begins with an ensemble of diverse, noisy samples in a high-dimensional space. The corrective vector field learned by

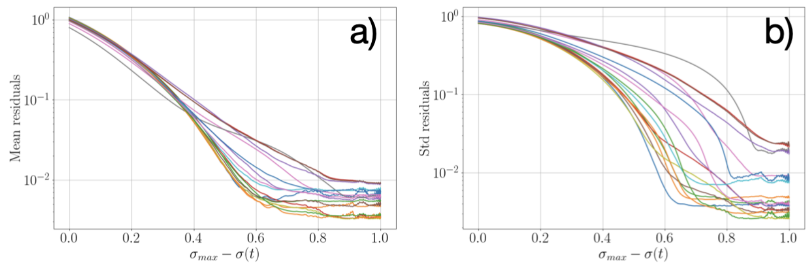

Figure 3: Evolution of residuals $e_t = y_t - y$ during the iterative refinement process. The PCDAE model follows the dedault hyperparameters and the Algorithms 1 and 2. (a) Mean of the absolute residuals and (b) Standard deviation of the absolute residuals for each output dimension, plotted against the denoising progress $(\sigma_{\max} - \sigma(t))$. Each colored line represents one of the 17 output dimensions.

the PCDAE then effectively guides this entire ensemble to converge towards the single, correct point on the physical manifold.

This behavior is analogous to the Flow Matching framework, where a learned vector field transports a simple prior distribution to a complex data distribution [14]. In our case, the PCDAE has successfully learned a field that deterministically maps a noisy distribution $(\mathcal{N}(0, I))$ to a single, sharp point, effectively inverting the diffusion process to find the unique solution always conditioned on the input condition x.

5 Conclusion

In this work, we introduced a progressive denoising autoencoder (PCDAE) as a highly efficient and robust surrogate model for complex physical systems that is more parameter- and data-efficient than a comparable Physics-Informed Neural Network (PINN) baseline. Moreover, we observe that our generative model, when trained on high-fidelity data, can learn physical laws so effectively that explicit, global constraints enforced at inference time become redundant and can even slightly hinder performance. Altogether, these findings validate the PCDAE framework as both effective and practically tunable for multi-output regression tasks.

Acknowledgments

This work was supported by the Portuguese FCT - Fundação para a Ciência e a Tecnologia, under project reference IDB/50010/2020 (https://doi.org/10.54499/UIDB/50010/2020), project reference UIDP/50010/2020 (https://doi.org/10.54499/UIDP/50010/2020) and project reference LA/P/0061/2020 (https://doi.org/10.54499/LA/P/0061/2020), and by the European Union under Horizon Europe project CAN-MILK (DOI:https://doi.org/10.3030/101069491). PV acknowledges support by project CEECIND/00025/2022 of FCT.

References

[1] Jascha Sohl-Dickstein et al. Deep unsupervised learning using nonequilibrium thermodynamics. *arXiv preprint arXiv:1503.03585*, Nov 2015.

[2] Pieter Abbeel Jonathan Ho, Ajay Jain. Denoising diffusion probabilistic models. *arXiv preprint arXiv:2006.11239*, Dec 2020.

[3] Stefano Ermon Yang Song. Generative modeling by estimating gradients of the data distribution. *arXiv preprint arXiv:1907.05600*, Oct 2020.

[4] Koray Kavukcuoglu Aaron van den Oord, Nal Kalchbrenner. Pixel recurrent neural networks. *arXiv preprint arXiv:1601.06759*, Aug 2016.

[5] A Tejero del-Caz et al. The lisbon kinetics boltzmann solver. *Plasma Sources Science and Technology*, 2019.

[6] Pedro Viegas et al. Surface recombination in pyrex in oxygen dc glow discharges: mesoscopic modelling and comparison with experiments. *Plasma Sources Science and Technology*, 2024.

[7] M. Raissi et al. Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. *Journal of Computational Physics*, 2019.

[8] Vasco Guerra Rodrigo Ventura Matilde Valente, Tiago C. Dias. Physics-consistent machine learning: output projection onto physical manifolds. *arXiv preprint arXiv:2502.15755*, Mar 2025.

[9] Paris Perdikaris Sifan Wang, Yujan Teng. Understanding and mitigating gradient pathologies in physics-informed neural networks. *arXiv preprint arXiv:2001.04536*, Jan 2020.

[10] Pascal Vincent et al. Extracting and composing robust features with denoising autoencoders. *Appearing in Proceedings of the 25th International Conference on Machine Learning*, 2008.

[11] Yoshua Bengio et al. Generalized denoising auto-encoders as generative models. *arXiv preprint arXiv:1305.6663*, Nov 2013.

[12] Yoshua Bengio Guillaume Alain. What regularized auto-encoders learn from the data generating distribution. *arXiv preprint arXiv:1211.4246*, Aug 2014.

[13] Ruiqi Gao et al. Learning energy-based models by diffusion recovery likelihood. *arXiv preprint arXiv:2012.08125*, Mar 2021.

[14] Yaron Lipman et al. Flow matching for generative model. *arXiv preprint arXiv:2210.02747*, Feb 2023.

[15] Yang Song et al. Score-based generative modeling through stochastic differential equations. *arXiv preprint arXiv:2011.13456*, Feb 2021.

[16] Robert Tibshirani Trevor Hastie and Jerome Friedman. *The Elements of Statistical Learning - Data Mining, Inference, and Prediction.* Springer Series in Statistics, 2008.

[17] Diederik P. Kingma et al. Variational diffusion models. *arXiv preprint arXiv:2107.00630*, Apr 2023.

[18] Fernando C. Pereira Naftali Tishby and William Bialek. The information bottleneck method. *arXiv preprint arXiv:physics/0004057*, Apr 2000.

[19] Ravid Shwartz-Ziv and Yann LeCun. To compress or not to compress - self-supervised learning and information theory: A review. *arXiv preprint arXiv:2304.09355*, Nov 2023.

[20] Luiz F. O. Chamon et al. Constrained sampling with primal-dual langevin monte carlos. *38th Conference on Neural Information Processing Systems (NeurIPS 2024)*, 2024.

[21] Ashish Vaswani et al. Attention is all you need. *arXiv preprint arXiv:1706.03762*, Aug 2023.

[22] Yuandi We et al. A review of physics-informed machine learning methods with applications to condition monitoring and anomaly detection. *arXiv preprint arXiv:2401.11860*, Jan 2024.

[23] Luiz F. O. Chamon et al. Constrained learning with non-convex losses. *IEEE TRANSACTIONS ON INFORMATION THEORY*, 2022.

[24] Neal Parikh and Stephen Boyd. Proximal algorithms. *Foundations and Trends in Optimization*, 2013.

[25] Stephen J. Wright Jorge Nocedal. *Numerical Optimization.* Springer, 200.

[26] Jimmy Lei Ba Diederik P. Kingma. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, Jan 2017.

A Derivation Physical-Constrained Inference

Now we present the full derivation of the updated refinement process (Eq. [10]) to enforce the physical constraints, $h(y) = 0$, at inference time.

We start by reinterpreting the iterative refinement update as the optimization problem:

$$y_{k,i+1} = y_{k,i} + \eta(g_{\sigma_k}(y_{k,i};x) - y_{k,i}) \implies \min_y F(y)$$
$$F(y) = \frac{1}{2}||y - (y_{k,i} + \eta(g_{\sigma_k}(y_{k,i};x) - y_{k,i}))||^2 \tag{11}$$

where $F(y)$ is seen as potential function that is minimized by the Eq.[6] at each refiinement step [24].

To enforce the physical constraints $h(y) = 0$ at each step, we have the constrained problem:

$$\min_y F(y) \text{ s.t. } h(y) = 0. \tag{12}$$

This is enforced in our iterative solver via an Augmented Lagrangian [20, 25]. We introduce a vector of dual variables $\lambda \in \mathbb{R}^m$ (one multiplier per constraint) and add both a Lagrange multiplier term and a quadratic penalty on the constraint violation. We define:

$$\mathcal{L}(y, \lambda) = F(y) + \lambda^T h(y) + \frac{\rho}{2}\|h(y)\|^2 \tag{13}$$

where λ collects the Lagrange multipliers enforcing $h(y) = 0$, and $\rho > 0$ is a penalty parameter that both stabilizes the dual updates and penalizes any remaining constraint violation.

The update for the primal variable y is derived from a gradient ($\nabla_y \mathcal{L}$) descent step, while the update for the dual variable λ is a gradient ($\nabla_\lambda \mathcal{L}$) ascent step [20]. This results in the following update rule for iteration step that replaces Eq.[6] in Algorithm 2:

$$y_k^{\tau+1} = y_k^\tau + \eta \left(g_{\sigma_k}(y_k^\tau x) - y_k^\tau\right) - \alpha \nabla h(y_k^\tau)^T (\lambda_k^\tau + \rho h(y_k^\tau)), \tag{14}$$

$$\lambda_k^{\tau+1} = \lambda_k^\tau + \beta\, h(y_k^{\tau+1}), \tag{15}$$

where τ corresponds to the inner iteration of the refinement level k (with σ_k). The first term in the y-update is exactly our unconstrained refinement step (with step-size η) , the second term corrects for constraint violation via the Jacobian ∇h, scaled by α, while the λ-update is a simple ascent on the dual variables with step-size β.

B PCDAE Model Architecture and Default Hyperparameters

The detailed layer-by-layer specification of our PCDAE model is provided in Table 1.

Table 1: Layer-wise architecture and default dimensions of the PCDAE model. B denotes the batch size.

Module	**Operation**	**Output Dimension**
	— Inputs —	
Input Y	Noisy Data $\vec{y}_\sigma$	$(B, \#y)$
Input X	Condition $\vec{x}$	$(B, \#x)$
Input σ	Noise Level σ	$(B, 1)$
	— Encoder / Projections —	
Noise Embedding	Sinusoidal Emb. + 2-Layer MLP	$(B, 10)$
Encoder Y Path	2-Layer MLP (GELU)	$(B, 58)$
Encoder X Path	2-Layer MLP (GELU)	$(B, 16)$
Fusion	Concatenate [Enc(Y), Enc(X), Emb(σ)]	$(B, 84)$
	— Decoder —	
Pre-processing	Layer Normalization	$(B, 84)$
Main Block	Linear $\rightarrow$ GELU	$(B, 58)$
Output Head	Linear Layer (predicts $\vec{y}$ and $\vec{x}$)	$(B, \#y + \#x)$

The default hyperparameters used for model training and inference are detailed in Table 2 and Table 3.

Table 2: Default training hyperparameters.

Parameter	**Value**
Noise Configuration	
Noise Kernel ($q_\sigma(\cdot \mid y)$)	Isotropic Gaussian $\mathcal{N}(y, \sigma^2\mathbf{I})$
Noise Level Distribution (p_σ)	Log-Uniform
Min Noise Level (σ_{min})	10^{-7}
Max Noise Level (σ_{max})	1.0
Max Vector Norm ($\epsilon_{\max}$)	0.5
Optimizer Configuration	
Optimizer	Adam [26]
Learning Rate	1×10^{-4}
Batch Size	32
Training Epochs	4500

Table 3: Default hyperparameters for the iterative refinement at inference.

Parameter	**Value**
Unconstrained Inference	
Noise Schedule	Geometric (‘geomspace‘)
Number of Noise Levels (K)	20
Steps per Level (T)	800
Generative Step Size (η)	0.01
Extra hyperparameters for *Constrained Inference*	
Primal Step Size (α)	10^{-4}
Dual Step Size (β)	10^{-2}
Penalty Parameter (ρ)	10^{-2}