# SoftFlow-DDN: Adaptive Flow and Soft Binning for Free-Form Conditional Density Estimation

ChengLong Song[1], Mazharul Islam[2], Bing Chen[3], Lin Wang[1,4], Shuangrong Liu[1,†], Bo Yang[4,1]
[1]Shandong Key Laboratory of Ubiquitous Intelligent Computing, University of Jinan, Jinan 250022, China
[2]Shandong Provincial Key Laboratory of Green and Intelligent Building Materials, University of Jinan, Jinan 250022, China
[3]David R. Cheriton School of Computer Science, University of Waterloo, Waterloo N2L3E8, Canada
[4]Quan Cheng Laboratory, Jinan 250100, China

***Abstract*—Free-form conditional density estimation (CDE) models arbitrary distributions without parametric constraints. Deconvolutional Density Networks (DDN) achieve this through discretizing the target space, yet they frequently produce spiked and degenerate densities. This failure arises from gradient isolation in hard binning—where only the target bin receives a learning signal—and is worsened by misalignment between the data distribution and the fixed bin grid. We present *SoftFlow-DDN*, a redesigned DDN that introduces two synergistic components: a *flow transformation*, which adaptively warps the target space to align with the grid, and a *soft binning* strategy, which distributes likelihood across adjacent bins via a temperature-controlled kernel. Together, they ensure global grid compatibility and local smoothness. Ablation studies confirm that both proposed components are necessary and complementary. Empirically, SoftFlow-DDN outperforms the original DDN and recent variants across toy and real-world tasks, achieving superior log-likelihood and producing visibly smoother, more realistic densities, particularly in low-data regimes.**

***Index Terms*—conditional density estimation, deconvolutional density network, data scarcity, flow-based transformation, soft binning**

## I. Introduction

Accurate modeling of the conditional distribution $p(y|\mathbf{x})$ of a target variable $y$ given a context $\mathbf{x}$ is fundamental to advance machine learning systems beyond deterministic point estimation ($E[y|\mathbf{x}]$). Although regression and classification focus on predicting expected values or discrete labels, many critical applications, including autonomous navigation, personalized healthcare, financial risk assessment, and scientific modeling, require a more enriched probabilistic understanding [1], [2]. In such settings, capturing heteroscedastic noise, multi-modality, and tail behavior is not merely beneficial but essential for safe and robust decision-making [3]–[5].

Conditional Density Estimation formally addresses this need, with the objective of inferring the complete density $p(y|\mathbf{x})$ from observed pairs $\{(\mathbf{x}_i, y_i)\}_{i=1}^N$. However, the problem remains particularly challenging when the underlying distribution family is unknown a priori, ruling out restrictive parametric forms. Classical parametric approaches (e.g., Gaussian mixtures) impose restrictive assumptions [6], while nonparametric methods (e.g., kernel density estimation) scale poorly with high-dimensional data [7], [8]. More recent deep approaches, particularly normalizing flows [9]–[12], offer high expressivity through invertible transformations and exact likelihood evaluation. Yet they often require complex architectures to maintain tractable Jacobians and tend to overfit or underperform when data are scarce [4].

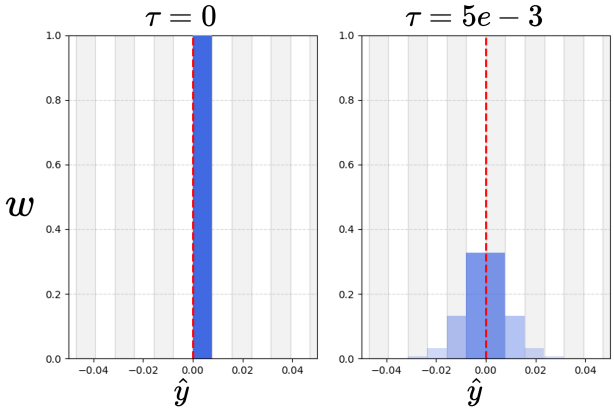


Fig. 1. Comparison between hard and soft binning strategies. The red-dashed line indicates the ground-truth position of the transformed target, $\hat{y} = 0$. **Left (Standard DDN):** The original DDN employs a hard binary search, assigning the target exclusively to a single bin (weight=1) while discarding all others. **Right (Ours):** The proposed *soft binning* strategy distributes weights to neighboring bins based on proximity.

This gap has motivated the development of deep learning-based "free-form" CDE methods, which leverage the expressive power of neural networks to model arbitrarily conditional

This work was supported in part by the National Natural Science Foundation of China under Grant Nos. 61872419, 62072213, and 62403209; in part by the Shandong Provincial Natural Science Foundation under Grant Nos. ZR2022JQ30, ZR2022ZD01, ZR2023LZH015, and ZR2024QF021; in part by the Shandong Provincial Key R&D Program Project under Grant Nos. 2024CXPT084 and 2025CXPT105; in part by the Key Research Project of Quan Cheng Laboratory, China under Grant Nos. QCL20250105 and QCL20250304; in part by the Open Fund of the State Key Laboratory of Silicate Materials for Architectures under Grant No. SYSJJ2025-02; in part by the New 20 Measures for Colleges and Universities of Jinan under Grant No. 202534127; in part by the Youth Innovation Team Program of Shandong Higher Education Institutions, China under Grant No. 2024KJH104; and in part by the University of Jinan Young Faculty Interdisciplinary Convergence Development Project 2025 under Grant No. XKJC-202507.
[†]Corresponding author: Shuangrong Liu (liusr.constant@gmail.com)

densities without pre-specified parametric forms [13]–[15]. Among these, Deconvolutional Density Network [16] stands out by discretizing the target space $\mathcal{Y}$ into $K$ bins with fixed edges $\{b_0, b_1, \ldots, b_K\}$ and modeling conditional density $p(y|\mathbf{x})$ as a piecewise-constant function:

$$p(y|\mathbf{x}) \approx \sum_{k=0}^{K-1} \pi_k(\mathbf{x}) \cdot \mathbb{I}_{[b_k, b_{k+1})}(y),$$

where $\pi_k(\mathbf{x}) \geq 0$, $\sum_{k=0}^{K-1} \pi_k(\mathbf{x}) = 1$, and $\mathbb{I}_{[b_k, b_{k+1})}$ is the indicator function for the $k$-th bin. The bin probabilities $\pi_k(\mathbf{x})$ are produced by a deconvolutional network, which first encodes $\mathbf{x}$ into a latent representation, then applies a variational layer for regularization, and finally decodes via deconvolutional layers to output log-masses $\mathbf{h} \in \mathbb{R}^K$ that are normalized via softmax.

Despite its flexibility, DDN suffers from two interrelated flaws that become acute in data-scarce regimes. First, the hard binning objective creates a gradient isolation problem: for each sample $(\mathbf{x}_i, y_i)$, only the bin $k^*$ containing $y_i$ receives a non-zero gradient, since $\mathbb{I}_{[b_k, b_{k+1})}(y_i) = 0$ for $k \neq k^*$. Consequently,

$$\frac{\partial \mathcal{L}_{\text{DDN}}}{\partial \pi_k(\mathbf{x}_i)} = \begin{cases} -\frac{1}{\pi_{k^*}(\mathbf{x}_i)} & \text{if } k = k^* \\ 0 & \text{otherwise,} \end{cases}$$

leaving adjacent bins untrained and producing degenerate, spiked densities lacking smoothness. This rigid assignment mechanism is visualized in Figure 1 *(left)*, which shows how a target point is exclusively assigned to a single bin, preventing gradient propagation to neighboring bins. Second, the fixed, uniform binning grid is fundamentally misaligned with real-world data distributions. If the true conditional density $p(y|\mathbf{x})$ concentrates mass in narrow regions, most bins receive near-zero probability, leading to inefficient parameter use and exacerbating overfitting, particularly in low-data regimes where the number of samples $N$ is much smaller than the number of bins $K$ [16].

In this study, we propose *SoftFlow-DDN*, a redesigned DDN architecture that overcomes both limitations through two synergistic components. First, we introduce a *flow transformation* (more details in section II-B) $f_\phi(\cdot;\theta)$ that warps the target space via an invertible mapping: $\hat{y} = f_\phi(y;\theta), \quad \theta = \psi(\mathbf{x})$, where $\psi$ is a parameter network conditioned on $\mathbf{x}$. This transformation aligns the warped density $p(\hat{y}|\mathbf{x})$ with the uniform binning grid, ensuring balanced gradient flow across bins [11]. Second, we replace hard binning with a *soft binning strategy* (more details in section II-C) that distributes the training signals across neighboring bins. For a temperature hyperparameter $\tau > 0$, we define soft weights $w_k(\hat{y})$ for each bin $k$, which provides gradients to all bins with non-zero weight, explicitly enforcing local smoothness. Our soft binning approach is illustrated in Figure 1 (right), which contrasts with the hard assignment by showing how weights are distributed to adjacent bins based on proximity, enabling smoother gradient flow and a more continuous density estimate. Moreover, the soft binning is crucial for stabilizing the joint training of the flow and the density estimator, as it provides a smooth loss landscape with respect to the flow parameters.

These components act together: the flow transformation optimizes global alignment between data and grid, while soft binning ensures local continuity and gradient sharing. We evaluate SoftFlow-DDN on a suite of toy and real-world tasks, demonstrating consistent improvements over original DDNs and recent variants. Our contributions are three-fold:

1) We identify and formalize gradient isolation and grid–distribution mismatch as root causes of degenerate behavior in DDN.
2) We propose SoftFlow-DDN, integrating a flow transformation for adaptive space alignment and a differentiable soft binning mechanism for gradient-aware smoothing.
3) We demonstrate superior performance across diverse toy and real-world CDE tasks, with notable gains in low-data regimes, and provide ablation studies confirming the necessity of both components.

The author will share the data/code available upon acceptance.

## II. Methodology

In this section, we first review the foundational framework of the Deconvolutional Density Network and identify its limitations regarding fixed discretization and sparse supervision. To address these issues, we propose two novel enhancements: a flow-based transformation to normalize the output space adaptively, and a soft binning strategy to smooth the optimization landscape.

### A. Preliminaries: Deconvolutional Density Network

The DDN aims to estimate the conditional density $p(y|\mathbf{x})$ by combining the merits of variational inference and deconvolutional decoding. The process can be summarized as follows:

**Encoding and Variational Layer.** Given an input-output pair $(\mathbf{x}, y)$, DDN employs an encoder $E(\mathbf{x})$ to map the condition $\mathbf{x}$ into a latent representation. To improve generalization and prevent overfitting, a Variational Layer (VL) is introduced [17]. It predicts a mean $\boldsymbol{\mu}(\mathbf{x})$ and variance $\boldsymbol{\sigma}^2(\mathbf{x})$, from which a latent variable $\mathbf{z}$ is sampled using the reparameterization trick:

$$\mathbf{z} = \boldsymbol{\mu}(\mathbf{x}) + \boldsymbol{\sigma}(\mathbf{x}) \odot \boldsymbol{\epsilon}, \quad \boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I}).$$

**Deconvolutional Estimation.** The latent variable $\mathbf{z}$ is fed into a Deconvolutional Estimator (decoder) [18]. Through successive up-sampling and convolution operations, the network outputs a log-mass vector $\mathbf{h} \in \mathbb{R}^K$ corresponding to $K$ discretized bins. The probability mass for each bin is obtained via Softmax:

$$\pi_k = \frac{\exp(h_k)}{\sum_j \exp(h_j)}.$$

**Density Retrieval.** To evaluate the likelihood of a target scalar $y$, DDN assumes a fixed domain partitioned into bins with edges $\{b_0, b_1, \ldots, b_K\}$. A binary search is performed to

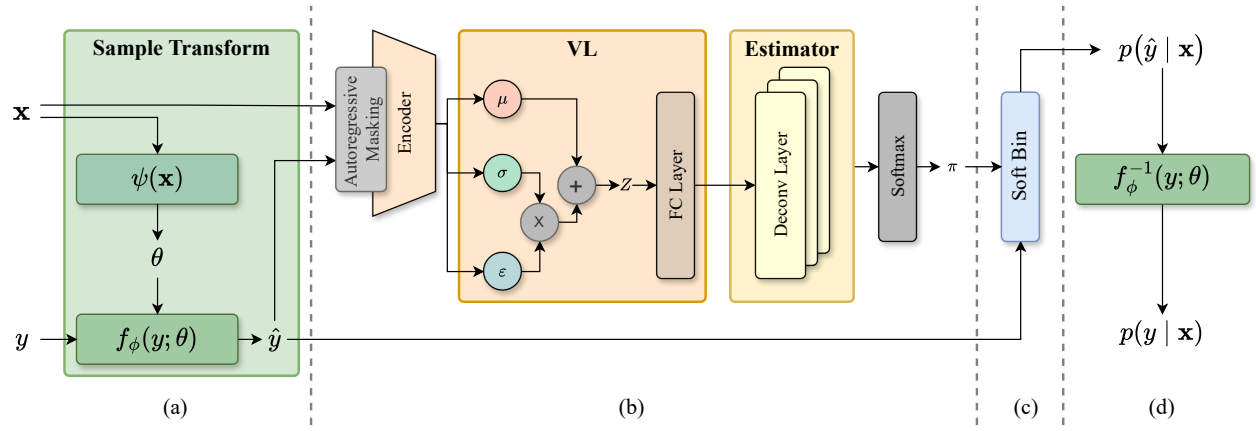


Fig. 2. Semantic architecture of SoftFlow-DDN (SF-DDN). The framework is composed of four integrated stages: **(a) Flow Transformation:** A parameter network $\psi(\mathbf{x})$ generates context-aware parameters $\theta$ to transform the raw target $y$ into a normalized latent representation $\hat{y}$, adapting the data geometry to the fixed grid (Eq. (2)). **(b) Deconvolutional Estimator:** The DDN backbone extracts features from $\mathbf{x}$ and predicts the raw log-mass distribution over the discretized bins. **(c) Soft Binning Strategy:** Instead of a hard assignment, this module computes soft weights $w_k$ based on the distance between $\hat{y}$ and bin boundaries using a learnable temperature $\tau$, ensuring differentiable gradient propagation. **(d) Density Recovery:** Corresponding to the inverse of (a), the final conditional density $p(y|\mathbf{x})$ is reconstructed by modulating the latent density $p(\hat{y}|\mathbf{x})$ with the Jacobian determinant of the flow transformation.

find the specific bin index $k^*$ such that $b_{k^*} \leq y < b_{k^*+1}$. The probability density is then approximated as:

$$p(y|\mathbf{x}) \approx \frac{\pi_{k^*}}{\Delta_k}, \tag{1}$$

where $\Delta_k = b_{k^*+1} - b_{k^*}$ is the bin width. For multi-dimensional outputs, DDN utilizes auto-regressive masking to capture dependencies between dimensions.

### B. Flow Transformation for Adaptive Domain

To mitigate the issue of non-uniform data distribution and inefficient bin usage, we introduce a flow transformation. Instead of forcing DDN to learn complex distributions on a fixed grid directly, we transform the target $y$ into a latent space $\hat{y}$ where the distribution is more uniform and suitable for the DDN's fixed discretization. This process is illustrated in Figure 2(a).

We define an invertible transformation function $f_\phi : \mathcal{Y} \to \hat{\mathcal{Y}}$, parameterized by $\theta$. To make this transformation conditional on $\mathbf{x}$, we employ a parameter network $\psi(\mathbf{x})$:

$$\theta = \psi(\mathbf{x}), \quad \hat{y} = f_\phi(y;\theta). \tag{2}$$

In our implementation, as detailed in Figure 3, $f_\phi$ is realized using a stack of Rational Quadratic Spline (RQS) flows [11], which offer high flexibility in modeling complex deformations. The parameter network $\psi(\mathbf{x})$ adopts a structure inspired by Variational Autoencoders (VAE) [17]. Specifically, the input $\mathbf{x}$ is first mapped by a shared encoder to a latent representation $\mathbf{z}_\theta$. Subsequently, independent lightweight decoders take $\mathbf{z}_\theta$ as input to generate the specific transformation parameters $\theta^{(l)}$ (i.e., bin widths, heights, and derivatives) for each layer $l$ of the flow network.

The DDN is then tasked with learning the density of the transformed variable, $p(\hat{y}|\mathbf{x})$. According to the change of variables formula, the density in the original space is recovered by:

$$p(y|\mathbf{x}) = p_(\hat{y}|\mathbf{x}) \cdot \left| \det \frac{\partial f_\phi(y;\theta)}{\partial y} \right|. \tag{3}$$

This hybrid approach allows the Flow to handle the global geometry and "warping" of the space, while the DDN focuses on modeling the local details in the regularized $\hat{y}$ space.

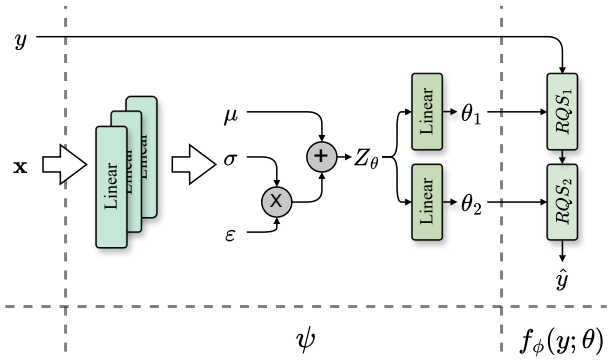


Fig. 3. Overview of the proposed flow transformations (Fig. 2(a)). To adaptively normalize the target space, we employ a parameter network $\psi$ with a VAE-like architecture. The condition $\mathbf{x}$ is first processed by a shared encoder to obtain a high-level feature representation $\mathbf{z}_\theta$. This latent code is then fed into separate, layer-specific decoders. Each decoder independently generates the transformation parameters $\theta$ (including widths, heights, and derivatives) required by the corresponding Rational Quadratic Spline (RQS) flow layer to map $y$ to $\hat{y}$.

### C. Soft Binning Strategy

To address the sparse gradient problem and avoid the limitations of "hard" assignments, we propose a soft binning strategy. Unlike standard approaches that rely on a fixed smoothing factor, we introduce a learnable temperature parameter ($\tau$).

This allows the model to dynamically balance the trade-off between focusing on the precise bin (high precision) and distributing gradients to neighboring bins (global supervision) during training.

**Soft Membership Computation.** For the transformed target $\hat{y}$ and a set of bin edges $\{b_k\}_{k=0}^{K}$, we first compute the signed distances to the left ($b_k$) and right ($b_{k+1}$) boundaries of each bin $k$:

$$D_{L,k} = \hat{y} - b_k, \quad D_{R,k} = b_{k+1} - \hat{y}.$$

To determine the relevance of each bin, we map these distances into a normalized activation space using the learnable temperature $\tau$. The boundary activations $\hat{D}_{L,k}$ and $\hat{D}_{R,k}$ are obtained via the sigmoid function $\sigma(\cdot)$:

$$\hat{D}_{L,k} = \sigma\left(\frac{D_{L,k}}{\tau}\right), \quad \hat{D}_{R,k} = \sigma\left(\frac{D_{R,k}}{\tau}\right).$$

Finally, the soft weight $w_k(\hat{y})$, representing the contribution of bin $k$ to the current target, is derived from the difference of these activations (detailed in Fig. 4):

$$w_k(\hat{y}) = \sigma\left(\hat{D}_{R,k} - \hat{D}_{L,k}\right). \tag{4}$$

This mechanism ensures that bins containing or near $\hat{y}$ receive higher weights, while distant bins retain small but non-zero gradients, preventing the "dead bin" phenomenon.

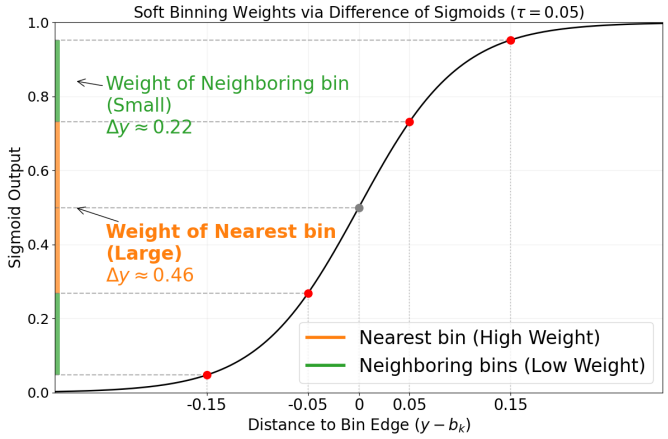


Fig. 4. Visualizing the computation of $w_k(\hat{y})$ Eq. (4) via boundary activations. The red points on the curve represent the bin boundaries, while the gray points represent $\hat{y}$. The target-enclosing bin (**orange**) attains high activation due to its proximity to $\hat{y}$. Simultaneously, the sigmoid function distributes gradient flow to adjacent bins (**green**) based on their distance.

**Density Estimation.** Instead of selecting a discrete log-mass from a single bin Eq. (1), we compute the expected log-mass for the target $\hat{y}$ via a weighted interpolation. Let $\pi_k(\mathbf{x})$ denote the predicted log-mass for the $k$-th bin. The interpolated log-mass $\tilde{\pi}(\hat{y}|\mathbf{x})$ and the final probability density are calculated as:

$$\tilde{\pi}(\hat{y}|\mathbf{x}) = \sum_{k=1}^{K} w_k(\hat{y}) \cdot \pi_k(\mathbf{x}), \quad p(\hat{y}|\mathbf{x}) = \frac{\exp(\tilde{\pi}(\hat{y}|\mathbf{x}))}{\Delta}, \tag{5}$$

where $\Delta$ is the uniform bin width in the transformed space.

**Temperature Regularization.** A critical challenge with a learnable $\tau$ is the model's tendency to collapse towards a trivial solution. To maximize the likelihood of the target bin, the optimization process naturally drives $\tau \to 0$, causing the soft weights to degenerate into a hard one-hot vector (Dirac delta). This reintroduces gradient sparsity and instability. To counteract this, we impose a penalty term that discourages the temperature from vanishing:

$$\mathcal{L}_\tau = -\lambda_\tau \log(\tau), \tag{6}$$

where $\lambda_\tau$ is a hyperparameter controlling the regularization strength. This loss term acts as an entropy constraint, encouraging the model to maintain a sufficient degree of "softness" in the bin assignments, thereby preserving the gradient flow across boundaries throughout the training process.

**Synergy with Flow Transformation.** The learnable soft binning strategy is particularly vital when coupled with the flow transformation. Since the target position $\hat{y} = f_\phi(y;\theta)$ in the latent space fluctuates as the flow parameters $\theta$ are updated, a hard assignment would cause the supervision signal to jump discontinuously between bins. The differentiable nature of our soft weighting, stabilized by the $\mathcal{L}_\tau$ regularization, ensures smooth gradient propagation back to the flow network, enabling robust joint optimization of the domain transformation and density estimation.

## III. Experiments

### A. Ablation Study

To empirically validate the individual contributions of our proposed modules, we conducted a component-wise ablation study on the *Elastic Ring* dataset. We evaluated five configurations to isolate the effects of the Flow Transformation and the Soft Bin Strategy against the baseline DDN with varying regularization strengths. Qualitative comparisons are visualized in Fig. 5.

**Analysis of Regularization ($\beta$).** Comparing the baseline DDN settings (rows 4 vs. 5), we observe that simply increasing the variance parameter $\beta$ reduces the error from 0.186 to 0.159 (≈15% improvement). While this alleviates the issue of "spike" distributions, it does so via *indiscriminate blurring*, washing out fine-grained topological details. In contrast, our SF-DDN achieves a significantly lower SSE (0.143) not by simple smoothing, but by structurally aligning the density, demonstrating a fundamental improvement over naive variance inflation.

**Impact of Flow Transformation (Topology).** Ablating the flow module (row 2) causes a sharp performance drop (SSE rises to 0.181, ≈26% drop). This highlights the limitation of a fixed grid in capturing non-linear manifolds. As observed qualitatively (Figure 5), without the flow warping, the density fails to conform to the ring's curvature. Specifically, at regions like $\mathbf{x} = 0.75$, the distribution exhibits abnormal *concave depressions* (inward collapse) where the fixed bins cannot adequately cover the data support. The Flow Transformation effectively resolves this by "warping" the space to match the grid.

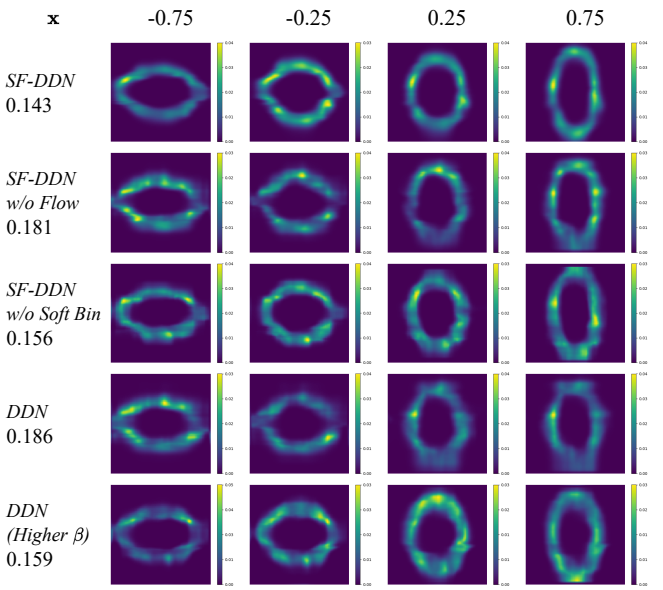


Fig. 5. Qualitative Ablation. (1) SoftFlow-DDN (SF-DDN): Captures the clean ring topology. (2) w/o Flow: Fails to align with the curvature. (3) w/o Soft Bin: Suffers from optimization instability (jagged artifacts). (4-5) DDN: Either too sharp ($\beta$ low) or over-blurred ($\beta$ high).

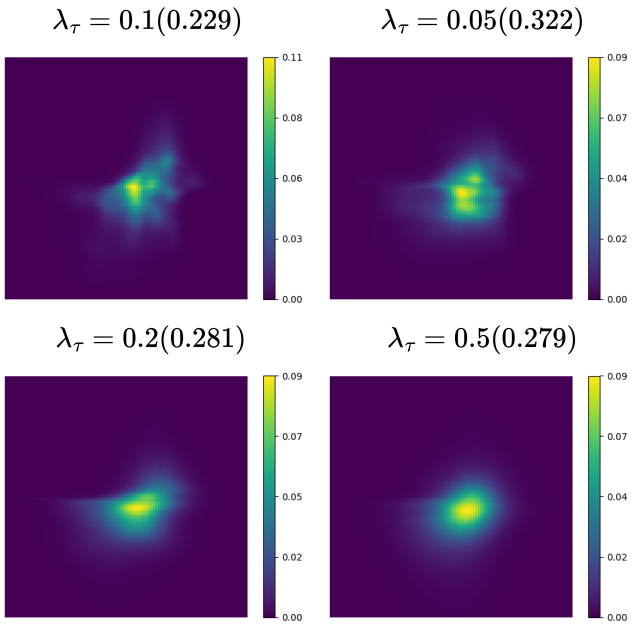


Fig. 6. Sensitivity to Regularization $\lambda_\tau$.The number in parentheses for each image indicates the corresponding SSE value. Optimal performance is found at $\lambda_\tau = 0.1$.

**Impact of Soft Bin Strategy (Stability).** Removing the soft binning (row 3, using flow + hard binning) yields an SSE of 0.156 (≈9% drop). Although better than the baseline, the resulting density profile remains "rugged" and discontinuous. This ruggedness arises because hard binning severs the gradient flow between adjacent intervals, preventing the flow network from converging to a smooth mapping. The soft binning strategy is thus critical: it provides the differentiable relaxation necessary to smooth the optimization landscape, bridging the gap between discrete binning and continuous flow training.

The full SF-DDN model achieves the best quantitative performance and the most topologically faithful density estimate, confirming that flow and soft binning are synergistic: the flow handles geometric alignment, while the soft binning ensures optimization stability.

### B. Hyperparameter Analysis

We investigate the regularization coefficient $\lambda_\tau$ (Eq. 6), which balances the "softness" of supervision against estimation precision. Fig. 6 shows the impact on SSE performance.

Results indicate a concave trade-off: *High* $\lambda_\tau$ (0.5) forces excessive smoothing, preventing the model from capturing sharp peaks. *Low* $\lambda_\tau$ (0.05) causes the mechanism to degenerate into quasi-hard binning, reintroducing gradient isolation and degrading performance. The optimal $\lambda_\tau = 0.1$ maintains sufficient gradient support for convergence while allowing precise density localization.

### C. Toy Conditional Density Estimation

*a) Datasets:* We evaluate on the four toy conditional density benchmarks used in DDN: *Squares*, *Half Gaussian*, *Gaussian Stick*, and *Elastic Ring*. To highlight the failure mode of hard bin assignment under limited supervision, we intentionally reduce the training set size to 500 samples for each toy task (1/4 of the 2000 samples used in the original DDN setup).

*b) Training protocol:* For a fair comparison, all methods are trained under the same data split and optimization budget. We select the checkpoint of each model by the *lowest validation SSE* during training, and report the corresponding evaluation performance.

*c) Metric:* Since the ground-truth conditional density $p^\star(y \mid \mathbf{x})$ is available for toy tasks, we measure the *sum of squared error (SSE)* between the predicted density $\hat{p}(y \mid \mathbf{x})$ and $p^\star(y \mid \mathbf{x})$ on a fixed grid $\{y_j\}_{j=1}^M$:

$$\mathrm{SSE} = \frac{1}{N}\sum_{i=1}^{N}\frac{1}{M}\sum_{j=1}^{M}\Big(\hat{p}(y_j \mid \mathbf{x}_i) - p^\star(y_j \mid \mathbf{x}_i)\Big)^2, \qquad (7)$$

where $\{(\mathbf{x}_i)\}_{i=1}^N$ are evaluation conditions and $\{y_j\}_{j=1}^M$ is a uniformly spaced grid over the target domain.

*d) Results:* Fig. 7 summarizes the SSE across the four toy tasks. Our method achieves consistently lower SSE on all tasks, with relative improvements of 9–27% over DDN (e.g., 27% on *Half Gaussian*, 21% on *Elastic Ring*). This demonstrates enhanced fidelity to the ground-truth density under extreme data scarcity. Qualitatively, the improvement aligns with our motivation: by replacing hard binning with soft supervision, our approach effectively mitigates the spiky artifacts that plague standard DDN in low-sample regimes.

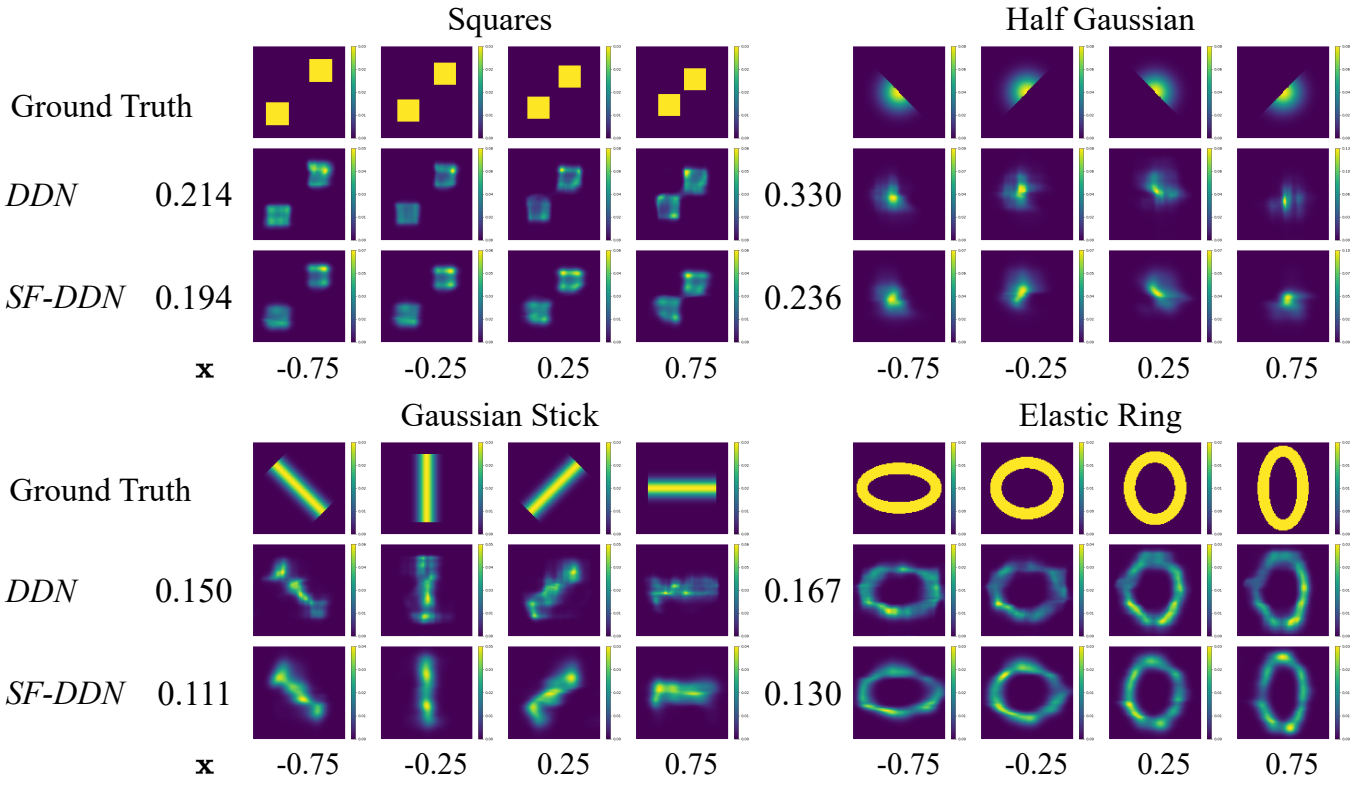


Fig. 7. Toy benchmarks (500 training samples). We report SSE (Eq. 7) on Squares, Half Gaussian, Gaussian Stick, and Elastic Ring. Our method yields lower SSE across all four datasets, suggesting reduced spikiness and improved density smoothness under limited samples.

### D. Real-World Datasets

We evaluate SoftFlow-DDN on five UCI datasets (Fish, Concrete, Energy, Parkinsons, Temperature) [19]. To empirically test performance in *data-scarce regimes*, we use a 10% training split with 10-fold cross-validation. We compare against MDN, Flow-based models (NSF, RNF), and standard DDN.

As shown in Table I, SoftFlow-DDN achieves the best NLL on 4 out of 5 datasets. Crucially, SoftFlow-DDN significantly outperforms the baseline DDN (e.g., reducing NLL from 0.97 to 0.74 on *Concrete*). This empirical gain validates that our *soft binning* and *flow transformation* effectively mitigate the gradient isolation problem inherent to fixed grids, especially when supervision is sparse. Furthermore, while pure flow models (NSF) tend to overfit in this low-data regime, SoftFlow-DDN strikes a superior balance between expressivity and generalization. Although DDN performs better on Parkinsons, SoftFlow-DDN's consistent superiority across diverse physics and environmental tasks confirms its reliability as a general-purpose CDE solver.

TABLE I
**TEST NLL ON UCI DATASETS (10% TRAINING DATA).** SOFTFLOW-DDN(SF-DDN) OUTPERFORMS BASELINES ON 4/5 TASKS, DEMONSTRATING ROBUSTNESS IN LOW-DATA SETTINGS. MEAN ± STD OVER 10 FOLDS.

| Model | Fish | Conc. | Energy | Park. | Temp. |
|---|---|---|---|---|---|
| NSF | 2.70±0.73 | 3.38±0.86 | 2.77±1.63 | 1.86±0.24 | 1.69±0.17 |
| RNF | 1.41±0.04 | 1.45±0.07 | 2.05±0.37 | 1.98±0.20 | 2.05±0.11 |
| MDN | 1.04±0.06 | 1.02±0.08 | -0.20±0.25 | 1.74±0.09 | 1.28±0.04 |
| DDN | 1.09±0.08 | 0.97±0.07 | 0.01±0.15 | **0.87±0.08** | 1.14±0.03 |
| **SF-DDN** | **1.01±0.07** | **0.74±0.08** | **-0.28±0.16** | 1.40±0.02 | **0.98±0.03** |

## IV. CONCLUSION

We presented SoftFlow-DDN, a principled enhancement of Deconvolutional Density Networks that addresses two fundamental failure modes in data-scarce conditional density estimation: *(i)* gradient isolation from hard binning, and *(ii)* grid–distribution mismatch due to fixed discretization. By integrating a conditional flow transformation for adaptive space warping and a temperature-controlled soft binning strategy for differentiable supervision, our method ensures both global alignment and local smoothness. Empirical results across toy and real-world tasks demonstrate consistent improvements over DDN and its recent variants—particularly in low-data regimes. The ablation study confirms that both components are necessary for robust and high fidelity density estimation.

**Limitations and Future Work.** Our method inherits the discretization requirement of DDN, which may limit resolution in very high-dimensional output spaces. The flow transformation, while flexible, adds computational overhead. Future directions include: *1)* exploring adaptive bin widths to

dynamically allocate resolution, *2)* developing more efficient flow architectures to reduce training cost, *3)* extending the framework to discrete-continuous hybrid domains, and *4)* providing theoretical analysis on the convergence and smoothness guarantees of the soft binning mechanism.

## REFERENCES


[1] A. Fanfarillo, B. Roozitalab, W. Hu, and G. Cervone, "Probabilistic forecasting using deep generative models," 2019.

[2] Y. Xue and C. Wu, "Financial crisis prediction of listed companies based on optimized bp neural networks," in *Proceedings of the 2023 7th International Conference on Cloud and Big Data Computing*, ser. ICCBDC '23. New York, NY, USA: Association for Computing Machinery, 2023, p. 61–67.

[3] B. Lakshminarayanan, A. Pritzel, and C. Blundell, "Simple and scalable predictive uncertainty estimation using deep ensembles," in *Advances in Neural Information Processing Systems*, I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, Eds., vol. 30. Curran Associates, Inc., 2017.

[4] B. L. Trippe and R. E. Turner, "Conditional density estimation with bayesian normalising flows," 2018.

[5] C. Winkler, D. Worrall, E. Hoogeboom, and M. Welling, "Learning likelihoods with conditional normalizing flows," 2023.

[6] C. Bishop, "Mixture density networks," Aston University, WorkingPaper 4288, 1994.

[7] A. B. Tsybakov, *Introduction to Nonparametric Estimation*. Springer, 2008.

[8] B. W. Silverman, *Density Estimation for Statistics and Data Analysis*. Chapman and Hall, 1986.

[9] D. J. Rezende and S. Mohamed, "Variational inference with normalizing flows," in *Proceedings of the 32nd International Conference on International Conference on Machine Learning - Volume 37*, ser. ICML'15. JMLR.org, 2015, p. 1530–1538.

[10] L. Dinh, J. Sohl-Dickstein, and S. Bengio, "Density estimation using real NVP," *CoRR*, vol. abs/1605.08803, 2016.

[11] C. Durkan, A. Bekasov, I. Murray, and G. Papamakarios, "Neural spline flows," 2019.

[12] J. Köhler, A. Krämer, and F. Noé, "Smooth normalizing flows," 2021.

[13] A. v. d. Oord, N. Kalchbrenner, O. Vinyals, L. Espeholt, A. Graves, and K. Kavukcuoglu, "Conditional image generation with pixelcnn decoders," in *Proceedings of the 30th International Conference on Neural Information Processing Systems*, ser. NIPS'16. Red Hook, NY, USA: Curran Associates Inc., 2016, p. 4797–4805.

[14] G. Papamakarios, T. Pavlakou, and I. Murray, "Masked autoregressive flow for density estimation," 2018.

[15] R. Li, B. J. Reich, and H. D. Bondell, "Deep distribution regression," *Computational Statistics & Data Analysis*, vol. 159, p. 107203, 2021.

[16] B. Chen, M. Islam, L. Wang, J. Gao, and J. Orchard, "Deconvolutional density network: Free-form conditional density estimation," *ArXiv*, vol. abs/2105.14367, 2021.

[17] D. P. Kingma and M. Welling, "An introduction to variational autoencoders," *Found. Trends Mach. Learn.*, vol. 12, no. 4, p. 307–392, Nov. 2019.

[18] J. Long, E. Shelhamer, and T. Darrell, "Fully convolutional networks for semantic segmentation," 2015.

[19] D. Dua and C. Graff, "Uci machine learning repository," 2017.