

NEURIPS 2026 · POSTER

DiffPTS: Rethinking Diffusion ELBO for Probabilistic Time Series Forecasting

Weiwei Ye¹ · Dongyuan Li¹ · Hangchen Liu¹ · Haotong Jiang² · Yoshihide Sekimoto¹ · Renhe Jiang¹

¹The University of Tokyo · ²Harbin Institute of Technology, Shenzhen

Paper arxiv.org/abs/2609.32363 · Code github.com/www155/DiffPTS

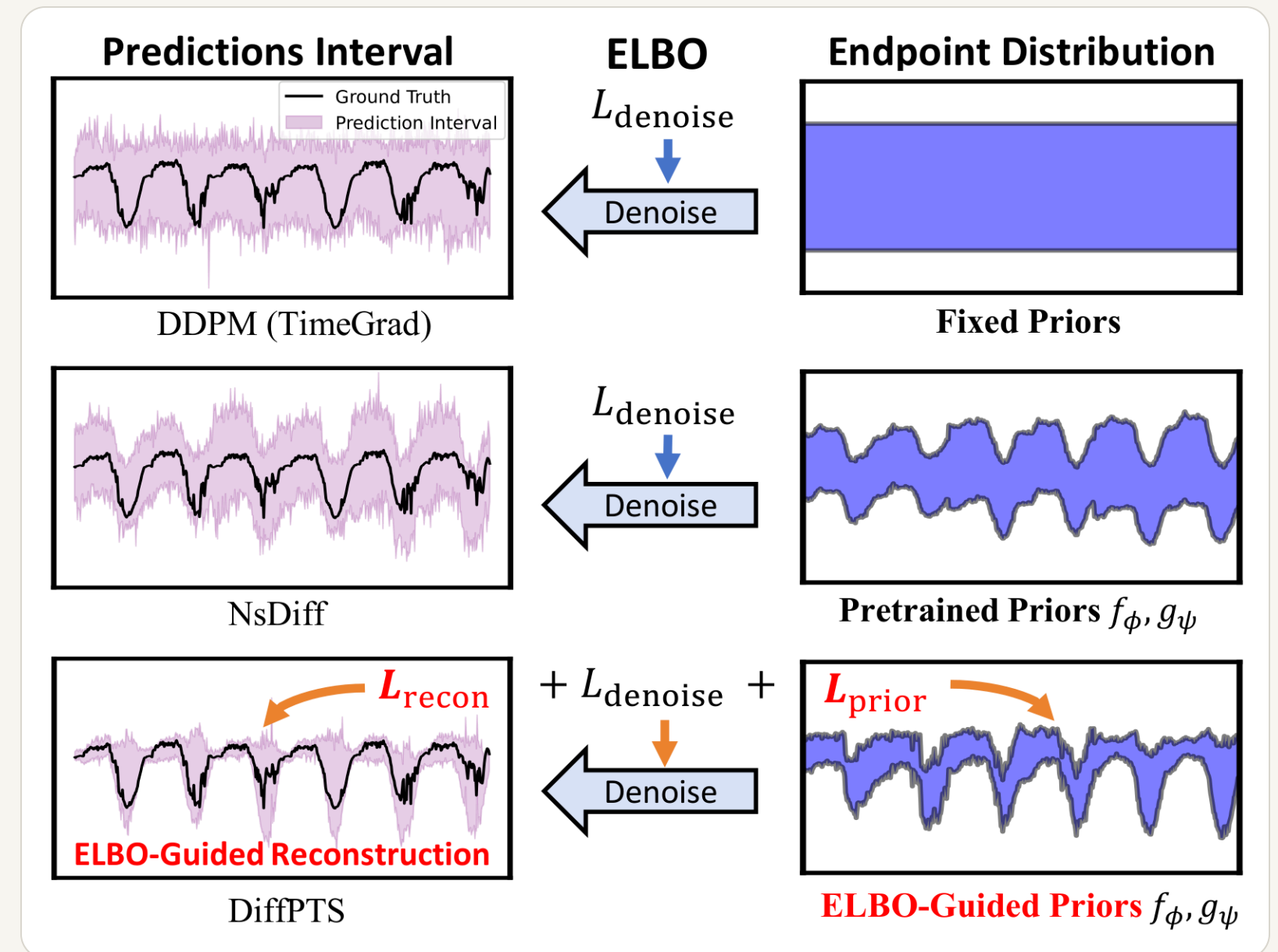


Forecasting under distribution shift

Probabilistic forecasting models $p(Y | X)$, the distribution of future values Y given the history X , and that distribution drifts over time.

Recent diffusion forecasters add a mean estimator $f(X)$ and a variance estimator $g(X)$, so the reverse process starts from an adaptive endpoint $N(f(X), g(X))$.

But f and g are pretrained with hand-designed targets, outside the diffusion ELBO. NsDiff's sliding-window variance, for example, over-estimates the uncertainty (right).



How existing diffusion forecasters train f and g

Model	Diffusion target	Endpoint	How f and g are trained
DDPM (TimeGrad, CSDI, TimeDiff)	Y	$N(0, I)$	no estimators
TMDM	Y	$N(f(X), I)$	f pretrained with MSE
D3U	$Y - f(X)$	$N(0, I)$	f pretrained with MSE
RDIT	$(Y - f(X)) / \sigma$	$N(0, I)$	f with MSE, σ from training residuals
NsDiff	Y	$N(f(X), g(X))$	f with MSE, g on sliding-window variance

Each design adds estimators to the diffusion process, yet trains them with its own regression target, decoupled from the variational objective.

THE MISSING PUZZLE

How should $f(X)$ and $g(X)$ be trained so that they align with the diffusion ELBO?

Once the endpoint carries learnable parameters, the ELBO terms that DDPM treats as constants stop being constant, and they must be optimized.

Rethinking the ELBO under the LSNM

LEARNABLE ENDPOINT AND FORWARD PROCESS

$$p(\mathbf{Y}_T \mid \mathbf{X}) = \mathcal{N}(f_\phi(\mathbf{X}), g_\psi(\mathbf{X}))$$

$$\mathbf{Y}_t = \sqrt{\bar{\alpha}_t} \mathbf{Y}_0 + (1 - \sqrt{\bar{\alpha}_t}) f_\phi(\mathbf{X}) + \sqrt{(1 - \bar{\alpha}_t)} g_\psi(\mathbf{X}) \epsilon_0$$

PROPOSITION 3.1 · THE FULL ELBO

$$-\text{ELBO} = C + \underbrace{\sum_{t=1}^T \gamma_t \mathbb{E} \|\epsilon_0 - \epsilon_\theta(\mathbf{Y}_t, \mathbf{X}, t)\|^2}_{\text{denoising}} + \underbrace{\frac{\bar{\alpha}_T}{2} \left\| \frac{f_\phi(\mathbf{X}) - \mathbf{Y}_0}{\sqrt{g_\psi(\mathbf{X})}} \right\|^2 + \sum_{i=1}^d \frac{\log [g_\psi(\mathbf{X})]_i}{2}}_{\text{constants in DDPM, trainable here}}$$

The constants γ_t and $\bar{\alpha}$ come from the noise schedule, so nothing is tuned. With $T = 20$ the endpoint weight $\bar{\alpha}$ is about 0.82, so the orange terms carry real weight in the bound.

The ELBO prescribes a Gaussian NLL for f and g

$$\mathcal{L}_{\text{ELBO}} = \underbrace{\mathbb{E}_{t, \epsilon_0} \left\| \epsilon_0 - \epsilon_{\theta}(\mathbf{Y}_t, \mathbf{X}, t) \right\|^2}_{\mathcal{L}_{\text{denoise}}(\theta, \phi, \psi)} + \underbrace{\frac{1}{2} \left\| \frac{f_{\phi}(\mathbf{X}) - \mathbf{Y}_0}{\sqrt{g_{\psi}(\mathbf{X})}} \right\|^2 + \frac{1}{2} \sum_{i=1}^d \log [g_{\psi}(\mathbf{X})]_i}_{\mathcal{L}_{\text{NLL}}(\phi, \psi): \text{ Gaussian NLL}}$$

Joint and end-to-end

θ , ϕ and ψ are trained together, with no pretraining stage.

Label-free variance

$g(\mathbf{X})$ is learned by the NLL itself, with no sliding-window or residual targets.

Generalizes the MSE

With $g \equiv 1$ the NLL reduces to the MSE that D3U and TMDM add.

One ELBO unifies existing paradigms

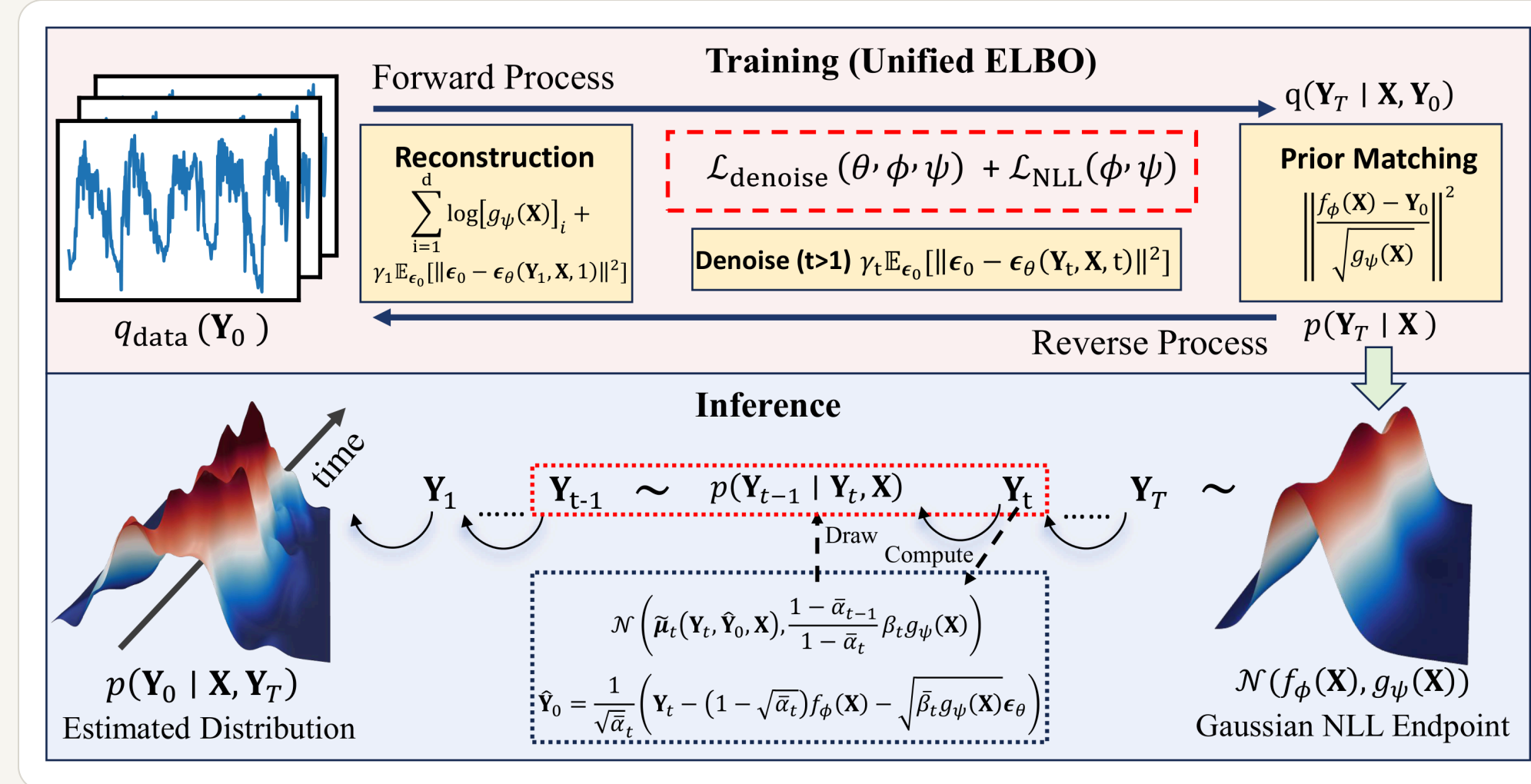
$$\text{ELBO}_{\text{NsDiff}} \xrightarrow{g_\psi = \sigma_{\text{trn}}} \text{ELBO}_{\text{RDIT}} \xrightarrow{g_\psi = 1} \text{ELBO}_{\text{D3U, TMDM}} \xrightarrow{f_\phi = 0} \text{ELBO}_{\text{DDPM}}$$

Under a shared standard noise schedule, residual-based methods (RDIT, D3U) and endpoint-based methods (NsDiff, TMDM) optimize the same ELBO with different choices of f and g .

So the DiffPTS objective plugs into any of them. Keep the host architecture, drop its pretraining and auxiliary losses, and train all parts jointly.

NsDiff's uncertainty-aware schedule relies on a supervised variance label, so it is complementary to this label-free objective.

DiffPTS: end-to-end training and sampling



Training

1. Draw a step t and noise ϵ_0 .
2. Build $\mathbf{Y}_t$ from $\mathbf{Y}_0$, $f(\mathbf{X})$ and $g(\mathbf{X})$.
3. Minimize the denoising loss plus the Gaussian NLL over θ , ϕ and ψ jointly.

Sampling

1. Draw the endpoint from $\mathcal{N}(f(\mathbf{X}), g(\mathbf{X}))$.
2. Denoise step by step with the learned reverse transitions.
3. Also runs with DPM-Solver in 5 steps.

State-of-the-art on nine benchmarks

CRPS	TMDM	D3U	RDIT	NsDiff	DiffPTS
ETTh1	0.452	0.447	0.410	0.392	0.374
ETTh2	0.383	0.385	0.380	0.358	0.355
ETTh1	0.375	0.412	0.336	0.346	0.329
ETTh2	0.289	0.282	0.258	0.256	0.254
ECL	0.461	0.439	0.362	0.290	0.218
EXG	0.336	0.354	0.392	0.324	0.275
ILI	0.967	2.097	0.845	0.806	0.731
Solar	0.350	0.488	0.225	0.300	0.173
Traffic	0.557	0.507	0.495	0.378	0.270

14.53%

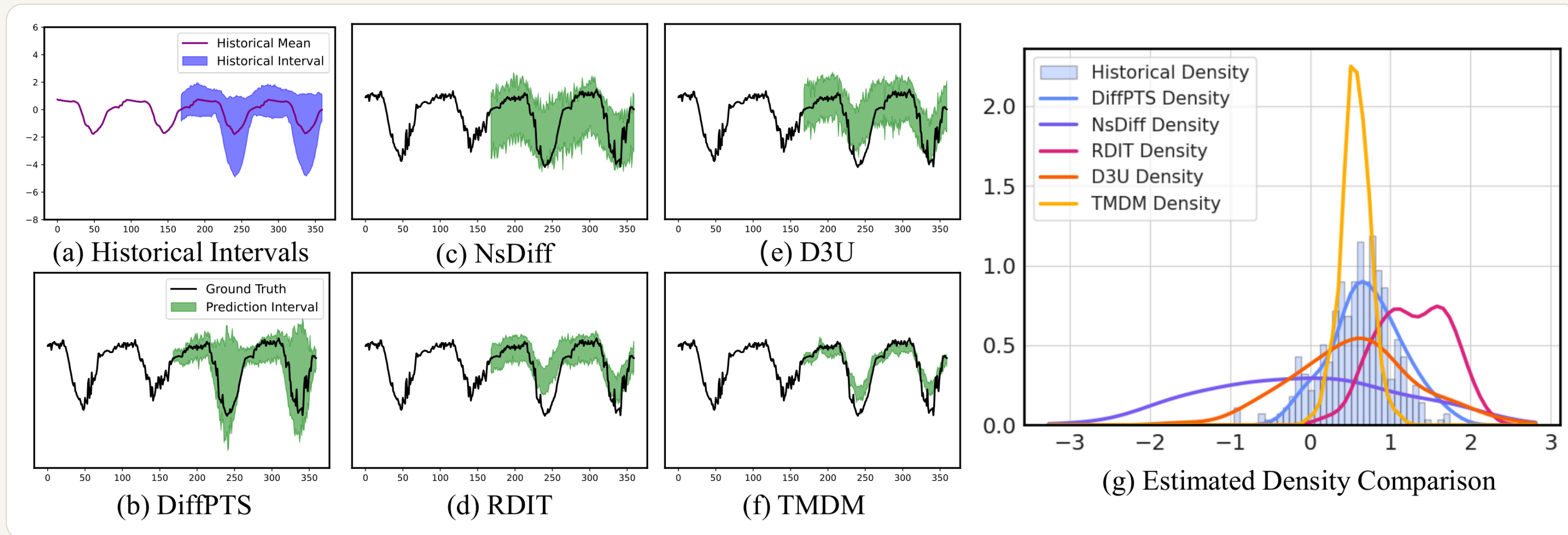
average CRPS reduction over existing
diffusion forecasters

16.55%

average MSE reduction

Lower CRPS is better. DiffPTS (blue) is best on
all nine datasets in both CRPS and MSE, also
against TimeGrad, CSDI, TimeDiff and
Diffusion-TS.

More faithful predictive distributions



An ETTm1 sample with 96% intervals. DiffPTS (b) follows the empirical intervals (a), while D3U and TMDM (e, f) keep nearly constant widths without a variance estimator. In (g), DiffPTS's density matches the historical density best.

A plug-and-play objective for existing models

Average CRPS reduction over nine datasets when each host keeps its architecture and is trained with the DiffPTS objective instead. Every host improves on all nine datasets.

TMDM

10.1%

lower CRPS

D3U

34.5%

lower CRPS

RDIT

15.7%

lower CRPS

NSDIFF

13.0%

lower CRPS

≈ 55%

less training time per epoch on ETTh1 for TMDM, D3U and RDIT. Joint training removes the pretraining stage and adds at most 8 MB of memory.

Ablations and analyses

Joint beats two-phase

With the same architecture, schedule and seeds, joint training lowers ETTh1 CRPS from 0.404 to 0.374 and MSE from 0.647 to 0.507.

Every ELBO term matters

ETTh1 CRPS is 0.374 with the full objective, 0.553 with f fixed to 0, 0.416 with g fixed to 1 and 0.437 without denoising.

Fast sampling

DPM-Solver with 5 steps stays within about 3% CRPS of 20-step ancestral sampling at about twice the speed.

Twenty steps suffice

$T = 20$ gives the best CRPS on all four datasets tested. $T = 1$ is only slightly worse, while $T = 1000$ degrades (ETTh1 CRPS 0.552).

Takeaways

1. With learnable endpoint estimators, the ELBO terms that DDPM drops become essential.
2. The ELBO prescribes a Gaussian NLL for f and g , so every component trains jointly, without pretraining.
3. One objective unifies and improves NsDiff, RDIT, D3U and TMDM.



Paper

arxiv.org/abs/2609.32363



Code

github.com/wwwy155/DiffPTS