LLM Flow Processes for Text-Conditioned Regression

Felix Biggs[*]
Secondmind
Wayve

Samuel Willis[*]
Secondmind

Abstract

Recent work has demonstrated surprisingly good performance of pre-trained LLMs on regression tasks (for example, time-series prediction), with the ability to incorporate expert prior knowledge and the information contained in textual meta-data. However we observe major error cascades even in short sequences $\lesssim 100$ points; these models are also computationally intensive and difficult to parallelise. Marginal LLM predictions do not suffer this issue and are trivially parallelised, but can predict over-broad densities. To address this, we propose combining these densities with a lightweight (flow-matching-based) neural process. We show that this combination leads to better-calibrated predictions overall, outputs locally consistent trajectories, and leads to text-conditioned function space selection in the meta-learner. As part of this work we propose a gradient-free (and non-Monte Carlo) method for sampling from a product-of-experts of a flow model and an 'expert' (here the LLM predictive densities). We believe this general method is of independent interest as it is applicable whenever an expert can be convolved with a Gaussian in closed form.

1 Introduction

Incorporating prior knowledge into regression models remains a fundamental challenge, particularly in domains where expert intuition is difficult to formalize. Pre-trained large language models (LLMs) integrate an enormous amount of existing information and can perform many new real-world tasks from very few examples [4]. Exciting new approaches are applying the implicit prior expert knowledge of these models to classical statistical tasks like regression, with some success [7, 15, 59, 63, 50]. Among these, the LLM-Process (LLMP; 50) can not only learn few-shot regression models, but can improve predictions with additional textual conditioning. As an example, they demonstrate that predictions on a weather time series improve by adding the textual conditioning of "daily temperature", and further with the addition of "in Montreal".

However, this approach has drawbacks. Williams et al. [63] evaluate the LLMP and their own improved Direct Prompt method, and observe what they term 'significant failures' on certain tasks, with predictions off by more than 500%. We observe a further failure mode of autoregressive models in this scenario (see, for example, Figure 1) where increasing the number of points the LLM is predicting at (the horizon or precision in x) causes cascading failures. We believe this relates to exposure bias when conditioning on their own predictions, which can cause exponential decay in predictive accuracy [3, 32, 25]. Further, these models are computationally very expensive as the models are large and require sequential sampling of one point at a time, and the samples themselves depend on the order in which new points are predicted.

A complementary line of work tackles the prior-knowledge problem through meta-learning, for example the Neural Process [12, 11] approach, which learns a probabilistic distribution over regression functions from a set of 'representative' datasets. These datasets can be synthetically-generated using

[*]Equal contribution. Correspondence to `felix.biggs@wayve.ai` and `samuel.willis@secondmind.ai`.

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

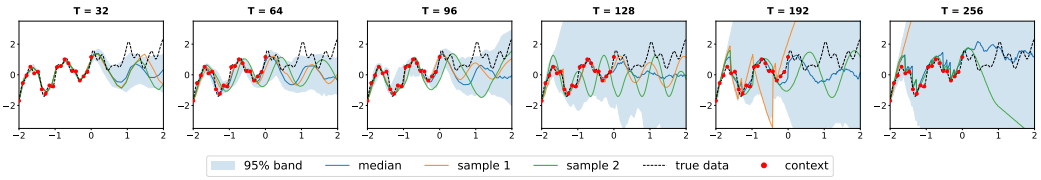

Figure 1: Predictions of the autoregressive Direct Prompt (*LLM-Direct*) method on a test function, conditioned on context data (red) and prompt "A periodic function with a linear trend", at different numbers of predictive points $T \in \{32, \ldots, 256\}$; prediction trajectories (used here as shorthand for joint samples, but not requiring time series-based methods) are unrolled 80 times. By $T = 96$ some divergent trajectories begin to appear and by $T = 256$ these are the default causing even a divergent median prediction. It is visible that these divergent error cascading trajectories often commit to a certain linear trend as a kind of attractor state, even when this disagrees with the data; at $T = 256$ some trajectories ignore all but the first few data points. We note similarity to the phenomenon of neural text degeneration [25, 65, 62, 61].

mechanisms (for example, causal graphs and augmentations) which mirror the processes generating real data. Impressive real-dataset results have been achieved with transfer from such synthetic data, for example by Müller et al. [40]. Diffusion- and flow-based generative modelling [54, 22, 57, 33] approaches to this setting like Neural Diffusion Processes (NDPs; 10) can learn intricate local structures while avoiding autoregression and being easy to parallelise [17]. Despite the power of this approach which enables us to learn arbitrary meta-learners through data, encoding expert knowledge about the specific domain (function space selection) of a *new* 'test' dataset remains difficult, and auxiliary meta-data which could inform models (for example natural language descriptions of variables) are instead ignored.

The weaknesses of the above approaches are complementary: autoregressive LLMs suffer sampling pathologies leading to a lack of robustness, while NDPs have no way to incorporate textual information. In order to combine the best strengths of each approach, we propose a new product-of-experts [21] sampling method to combine LLM Processes (with the ability to condition on prior world knowledge and rich, unstructured text) with NDPs (which can meta-learn over many divergent prior sources of regression data). Our method effectively allows the LLM text to filter within a function space defined by an NDP 'prior'.

Our main contributions are as follows:

- We introduce a principled gradient-free (with respect to the expert) non-Monte-Carlo product-of-experts sampling method for flow and diffusion generative models; this method is generally applicable whenever the expert(s) convolve in closed-form with a Gaussian.
- LLM Flow Processes (LLM-FPs) are an instantiation of this idea: by intersecting the NDP's meta-learned functions with the LLM's text-conditioned probabilities, we subselect the NDP-plausible joint samples consistent with the text. Specifically, we show that our method integrates the information from the LLM to filter the NDP-defined function space prior.
- We empirically demonstrate failures of autoregressive LLM regression under increasing prediction resolution or horizon, and that LLM-FPs generate samples from their NDP generative distribution consistent with their textual conditioning.

Structure. In Section 2 we discuss prerequisite and related work in neural processes, NDPs, flow matching, and conditioning of flow models; Section 3 introduces our general gradient-free method for expert-guided flow sampling; Section 4 discusses its instantiation as LLM-FP, sampling regression functions with LLM expert advice; Section 5 describes our empirical evaluation methodology and results for these models, which are discussed in Section 6; Section 7 concludes with implications and future work. Code is available at https://github.com/sam-willis/LLM-flow-processes.

2 Background and Related Work

2.1 Meta-Learning of Probabilistic Regression

The goal of meta-learning and Neural Processes [12], which subsumes methods like NDPs [10] and LLMPs [50], is to obtain predictive distributions over new points conditioned on a set of context points (or 'dataset', hence the 'meta' modifier). Formally, we have a sequence of n context input points $\mathcal{X}^{\text{data}} = (\boldsymbol{x}^{\text{data},i})_{i=1}^{n}$ and corresponding output points $\mathcal{Y}^{\text{data}} = (\boldsymbol{y}^{\text{data},i})_{i=1}^{n}$ forming our dataset $\mathcal{D} = (\mathcal{X}^{\text{data}}, \mathcal{Y}^{\text{data}})$. We would like to predict m target outputs $\mathcal{Y} = (\boldsymbol{y}^{j})_{j=1}^{m}$ at their inputs $\mathcal{X} = (\boldsymbol{x}^{j})_{j=1}^{m}$.[2]

In the stochastic process framing, we assume some distribution over tasks, where each task corresponds to a particular draw of a function f from some underlying distribution, and we observe noisy realisations of this function in the form of datasets, with outputs $\boldsymbol{y} = f(\boldsymbol{x}) + \text{noise}$. We would like to directly elicit or (meta)-learn the predictive distribution

$$p(\mathcal{Y} \mid \mathcal{X}, \mathcal{D}) = \int p(\mathcal{Y} \mid \mathcal{X}, f)\, p(f \mid \mathcal{D})\, \mathrm{d}f.$$

In the Neural Process literature this predictive distribution is learned rather than specified. A single model is trained on a large number of datasets sampled from the task distribution (the 'outer' learning loop) to learn how to do in-context prediction. At test time, prediction on a new dataset is in-context conditioning on $\mathcal{D}$ (the 'inner' learning loop), with no updates to the model weights. Note that we may wish to learn with a parametrised distribution which works for any possible m and n,[3] and have the option to condition also on additional side-information, $\mathcal{C}$, like meta-data.

We follow this standard evaluation protocol, inherited from Dutordoir et al. [10] and Requeima et al. [50] below: a single trained NDP is used for all tasks in this paper and LLMs are not trained or fine-tuned in any way. However, we emphasise that the focus of our method is not the meta-learning itself but a method to condition existing meta-learned models on textual information $\mathcal{C}$ via pre-trained LLMs.

LLM Processes and Direct Prompt. LLM Processes (50; which build on previous work such as 7, 15, 59) elicit a predictive distribution like that above by repurposing LLM predictive probabilities. Additional contextual information can simply be incorporated into the prompt, utilising the 'expert knowledge' of the LLM. The context data $\mathcal{D}$ and target locations $\mathcal{X}$ are contained in the prompt.

Samples are drawn by unrolling the LLM decoder in an autoregressive way (termed A-LLMP in 50). Note that the predictive probabilities will depend on the order we choose, which makes full density estimation infeasible. This scenario is examined by Williams et al. [63] who introduce an improved Direct Prompt method with constrained decoding that significantly improves on LLMP results. However, they note in all such models a prevalence of what they term 'significant failures', where predictions are off by more than 500%. We additionally run into standard issues with autoregressive models (where the model is conditioning on its own imperfect outputs which do not resemble training data; 3): errors can compound, possibly exponentially with longer prediction horizons (see Fig. 1); predictions can worsen dramatically with the number of points we wish to predict. We note in particular a tendency of mode collapse to a kind of 'attractor state' of predicting linear trends, perhaps related to the neural text degeneration phenomenon or 'repetition curse' [25, 65, 62, 61]. LeCun [32] argues that these problems fundamentally limit decoder-based LLM models.

An alternative to these autoregressive models is the *independent marginal* formulation (I-LLMP), where we assume conditional independence of the target points. The probabilities of each $\boldsymbol{y}^{j}$ can then directly obtained through the output logits of the LLM, with joint probability via multiplication [50]. We note however this may not capture correlation in the evaluation points, and we observe that lack of reasoning about the local structure can lead to some strange outliers (see the LLM-Independent *median* in Fig. 2).

[2] Throughout, we use $i \in \{1, \ldots, n\}$ to index context points (where the outputs $\boldsymbol{y}^{\text{data},i}$ are observed) and $j \in \{1, \ldots, m\}$ to index target points (where the outputs $\boldsymbol{y}^{j}$ are predicted).

[3] This problem has close relationships to learning a distribution over functions or stochastic process hence the framing above. If we satisfy the conditions of the Kolmogorov Extension Theorem it can define a valid stochastic process (see Appendix A).

Neural Diffusion Processes. Dutordoir et al. [10] instead focuses on learning a neural process defined by a *diffusion* generative model [54, 22, 57] in the output variables $\mathcal{Y}$. Samples can then be drawn from the joint distribution via the learned reverse (generative) process. By using an architecture that is equivariant to permutation of inputs and outputs (for example, by using self-attention) this produces exchangeability (in the sense of the KET; Appendix A), enabling coherent draws. The original paper used a de-noising score matching loss [60] taking input variables $(\mathcal{X}^{\text{data}}, \mathcal{X})$ to learn a distribution over both target *and* context outputs $(\mathcal{Y}^{\text{data}}, \mathcal{Y})$; conditioning on context outputs was done via a Re-Paint style recycling of context points [37]. Here we update conditioning to a modern method (based on 55, 46, see Section 2.2; this also allows us to account for standard Gaussian measurement noise), a flow matching objective [1, 33, 35], and a more standard diffusion transformer (DiT; 43). We note similar works of Hamad and Rosenbaum [17] and Rozet and Louppe [51], which respectively define a flow-matching-based NDP with *context* output points as inputs rather than generated variables, and generate joint samples with a diffusion model.

Synthetic Data. A significant component of the Neural process literature looks at generating synthetic data for training models before transfer to real-world datasets [5, 38, 41, 68, 47, 44]. For example Dutordoir et al. [10] trained on Gaussian Process samples, and Hollmann et al. [23] trained on data generated using random causal graphs and special augmentations (with discrete labels). Müller et al. [40] notes that this is equivalent to learning to implicitly marginalise over a Bayesian posterior distribution with the prior defined by the data-generation process, as also examined by Dutordoir et al. [10] (see in particular their Figure 3). Improved data-generation procedures have been explored in subsequent work [24, 66], including adversarially-designed generators [64]. These approaches are complementary to our contribution and could be incorporated into our training pipeline to further strengthen the learned prior, though we do not pursue them here.

2.2 Diffusion and Flow Matching

Diffusion models [54, 22, 57] and their generalisation to flow matching (1, 33, 35; see 34 for a comprehensive introduction) provide a simulation-free learning objective for generative modelling via differential equations. We adopt the popular class of affine conditional flows interpolating from Gaussian noise $\boldsymbol{Y}_0 \sim p_0 = p_{\text{noise}}$ to data $p_{\text{data}} = p_1$,[4] which can be defined by the conditional probability paths (for some scheduler pair α_t, σ_t[5]):

$$p_{t|1}(\boldsymbol{y}|\boldsymbol{y}_1) = \mathcal{N}(\boldsymbol{y}|\alpha_t \boldsymbol{y}_1, \sigma_t^2 I).$$

The flow matching or de-noising score matching objective [60] can be used to learn a neural network approximation to the marginals $p_t(\boldsymbol{y}) = \int p_{t|1}(\boldsymbol{y}|\boldsymbol{y}_1) p_1(\boldsymbol{y}_1) \mathrm{d}\boldsymbol{y}_1$. We parametrise this network as a velocity field u_t^θ (our flow network); in the affine Gaussian case this implies a score network $s_t^\theta(\boldsymbol{y}_t) \approx \nabla_{\boldsymbol{y}_t} \log p_t(\boldsymbol{y}_t)$ via Eq. (5). These can be linearly converted between for sampling purposes, as can direct prediction (via Tweedie's formula) of the expected final output $\widehat{\boldsymbol{y}}_{1|t}(\boldsymbol{y}_t) \coloneqq \mathbb{E}[\boldsymbol{Y}_1 | \boldsymbol{Y}_t = \boldsymbol{y}_t]$; see Appendix B for more details.

Conditioning Flow Models. Using our learned flow, we can adapt the above to sampling from a conditional distribution $p_1(\boldsymbol{y}|\boldsymbol{z})$ for some conditioning variable $\boldsymbol{z}$ (for example, we could prespecify certain components of $\boldsymbol{y}$ for an infilling task). From Bayes' rule,

$$\nabla_{\boldsymbol{y}_t} \log p_t(\boldsymbol{y}_t|\boldsymbol{z}) = \nabla_{\boldsymbol{y}_t} \log p_t(\boldsymbol{y}_t) + \nabla_{\boldsymbol{y}_t} \log p_t(\boldsymbol{z}|\boldsymbol{y}_t).$$

Pokle et al. [46] and Song et al. [55] consider problems where the measurement model $p_{\text{meas}}(\boldsymbol{z}|\boldsymbol{y}_1)$ is linear in $\boldsymbol{y}_1$ plus Gaussian noise. Generalising a method of Chung et al. [9], they express

$$p_t(\boldsymbol{z}|\boldsymbol{y}_t) = \int p_{\text{meas}}(\boldsymbol{z}|\boldsymbol{y}_1)\, p_{1|t}(\boldsymbol{y}_1|\boldsymbol{y}_t) \mathrm{d}\boldsymbol{y}_1,$$

and substitute the approximation;

$$p_{1|t}(\boldsymbol{y}_1 \mid \boldsymbol{y}_t) \approx \mathcal{N}\left(\boldsymbol{y}_1; \widehat{\boldsymbol{y}}_{1|t}(\boldsymbol{y}_t), r_t^2 I\right). \tag{1}$$

where $\widehat{\boldsymbol{y}}_{1|t}(\boldsymbol{y}) \coloneqq \mathbb{E}[\boldsymbol{Y}_1 | \boldsymbol{Y}_t = \boldsymbol{y}]$ is linearly derived from and approximated using our flow model (see Eq. 6 in Appendix B), and r_t is a time-dependent noise value that can depend on the data, which

[4] We notate the generated variable as $\boldsymbol{y}$ or $\boldsymbol{Y}$, rather than the more usual $\boldsymbol{X}$, as it will also represent the dependent variable in regression as defined by an NDP.

[5] In practice, we use the conditional Optimal Transport (OT) path [33]: $\alpha_t = t$ and $\sigma_t = 1 - t$.

Pokle et al. [46] argue should be set to $r_t^2 = \sigma_t^2/(\sigma_t^2 + \alpha_t^2)$. Under this approximation the score correction can be written in closed form and used to guide flows towards the desired conditional. In practice because the measurement model is learned separately from the flow model, the calibration of the models may not match; an additional multiplicative weighting may be added to the conditioning score to compensate for this. Using the correspondence between the score and the flow (Eq. 5) this guidance for the score function can then be adapted into guidance via flow directly. We note further proposed methods for improving the performance of conditional sampling, such as using variance estimates in the approximation Eq. (1) [52], or aligning the prior density p_0 using Gaussian Processes [27].

Closely related work looks at sampling from energy tilted densities like $\propto p_1(\boldsymbol{y})e^{-\ell(\boldsymbol{y})}$, for energy (loss) ℓ [56, 67, 16, 53, 2, 28]. These works also use a guided score function for this combined density and make the same normal approximation for $p_{1|t}$, leading to an additional score term like $\nabla_{\boldsymbol{y}_t} \log \int \mathcal{N}\left(\boldsymbol{y}_1;\, \widehat{\boldsymbol{y}}_{1|t}(\boldsymbol{y}_t),\, r_t^2 I\right) e^{-\ell(\boldsymbol{y}_1)} \mathrm{d}\boldsymbol{y}_1$, which is then approximated in some way, for example through Monte Carlo sampling. This can be used for a variety of purposes, for example by Kong et al. [28] to optimise an objective ℓ on complex domains learned through the diffusion model while avoiding infeasible solutions like non-synthesizable molecules. A similar additional score term appears in our gradient-free product-of-experts method introduced in Section 3.

3 Gradient-Free Product-of-Experts Guidance for Flow and Diffusion Models

We utilise a products of experts [21] which is the natural way to combine two distributions encoding independent sources of information. We begin with a distribution p_1 defined by our flow model,[6] and q an "expert" distribution (for example LLMP predictions) we wish to integrate information from via an extra conditioning variable c (for example a textual prompt). We define the product of experts distribution of these as

$$\pi_1(\boldsymbol{y}_1|c) \coloneqq \frac{1}{Z_c} q(\boldsymbol{y}_1|c) p_1(\boldsymbol{y}_1)$$

where $Z_c = \int q(\boldsymbol{y}_1|c)p_1(\boldsymbol{y}_1)\mathrm{d}\boldsymbol{y}_1$ is a normalising constant. An advantage of this product-of-experts formulation is that density is concentrated where both q and p_1 have probability mass, so we are unlikely to end up in the low-density regions of p_1 where our flow model has poorly learned the underlying density.

Using the same conditional probability distributions as our original flow model (for example, the OT affine Gaussian flow), we define a time-dependent version of the product distribution,

$$\pi_t(\boldsymbol{y}_t|c) \coloneqq \int \pi_1(\boldsymbol{y}_1)\, p_{t|1}(\boldsymbol{y}_t \mid \boldsymbol{y}_1)\, \mathrm{d}\boldsymbol{y}_1 = \frac{1}{Z_c} \int q(\boldsymbol{y}_1|c) p_1(\boldsymbol{y}_1)\, p_{t|1}(\boldsymbol{y}_t \mid \boldsymbol{y}_1)\, \mathrm{d}\boldsymbol{y}_1.$$

In order to produce guided sampling we use Bayes' Theorem to rewrite the score function

$$\begin{aligned}
\nabla_{\boldsymbol{y}_t} \log \pi_t(\boldsymbol{y}_t) &= \nabla_{\boldsymbol{y}_t} \log \int q(\boldsymbol{y}_1|c)\, p_{t|1}(\boldsymbol{y}_t \mid \boldsymbol{y}_1)\, p_1(\boldsymbol{y}_1)\, \mathrm{d}\boldsymbol{y}_1 \\
&= \nabla_{\boldsymbol{y}_t} \log \int q(\boldsymbol{y}_1|c)\, p_{1|t}(\boldsymbol{y}_1 \mid \boldsymbol{y}_t)\, p_t(\boldsymbol{y}_t)\, \mathrm{d}\boldsymbol{y}_1 \\
&= \nabla_{\boldsymbol{y}_t} \log p_t(\boldsymbol{y}_t) + \nabla_{\boldsymbol{y}_t} \log \int q(\boldsymbol{y}_1|c)\, p_{1|t}(\boldsymbol{y}_1 \mid \boldsymbol{y}_t)\, \mathrm{d}\boldsymbol{y}_1.
\end{aligned}$$

This closely parallels the conditional score derivation in Section 2.2, with the expert $q(\cdot|c)$ playing the role of the measurement model. We also adopt the approximation of $p_{1|t}(\boldsymbol{y}_1 \mid \boldsymbol{y}_t) \approx \mathcal{N}\left(\boldsymbol{y}_1;\, \widehat{\boldsymbol{y}}_{1|t}(\boldsymbol{y}_t),\, r_t^2 I\right)$ as in Eq. (1), to find

$$\int q(\boldsymbol{y}_1|c)\, p_{1|t}(\boldsymbol{y}_1 \mid \boldsymbol{y}_t)\, \mathrm{d}\boldsymbol{y}_1 \approx \int q(\boldsymbol{y}_1|c)\, \mathcal{N}\left(\boldsymbol{y}_1;\, \widehat{\boldsymbol{y}}_{1|t}(\boldsymbol{y}_t), r_t^2 I\right)\, \mathrm{d}\boldsymbol{y}_1 =: \widetilde{q}_{r_t}(\widehat{\boldsymbol{y}}_{1|t}(\boldsymbol{y}_t)|c) \qquad (2)$$

[6] For simplicity in this section we consider the standard setting of a flow over a single variable $\boldsymbol{y}_t$, rather than the more complex NDP setup.

also using $\widehat{\boldsymbol{y}}_{1|t}(\boldsymbol{y}) \coloneqq \mathbb{E}[\boldsymbol{Y}_1|\boldsymbol{Y}_t = \boldsymbol{y}]$ as defined in Section 2.2. Similarly, r_t is chosen in some way based on the data, for example following Pokle et al. [46]. Sampling is then possible through the score formulation of affine flows via

$$\nabla_{\boldsymbol{y}_t} \log \pi_t(\boldsymbol{y}_t|c) \approx s_t^\theta(\boldsymbol{y}_t) + \nabla_{\boldsymbol{y}_t} \log \widetilde{q}_{r_t}(\widehat{\boldsymbol{y}}_{1|t}|c). \tag{3}$$

$\tilde{q}$ can be rewritten as a convolution with a zero-mean Gaussian, and hence corresponds to a density, which is evaluated at $\widehat{\boldsymbol{y}}_{1|t}$.[7] Specifically, it is the density of draws from q corrupted by Gaussian noise of scale r_t - a quantity arising commonly in signal processing (for example, as a "Gaussian filter") or in kernel density estimation. $\widetilde{q}_{r_t}$ is infinitely differentiable, and its score function can be computed without *taking gradients of* q, only through the flow network via $\widehat{y}_{1|t}$; in this sense our method is *gradient free*.

Up to Eq. (3) our method is similar to the conditional score [46, 55] or to energy-tilted methods [56, 67, 16, 53, 2] which use Monte Carlo estimation for the second term. However unlike these works we will instead work with distributions where the convolution is actually tractable, noting that a large such class exists. For example, in the case of an LLM (as considered below) convolution is trivial because it outputs probabilities in bins (see Eq. 4). This convolution trick avoiding gradients of q is a core contribution, and could be extended in other settings: for example, all piecewise polynomial functions have closed form convolutions with Gaussians using only polynomials and Φ; this takes us into the field of spline smoothing [58].

4 LLM-Flow Processes

We use an NDP as defined in Section 2.1 to model a joint distribution, and add product-of-expert conditioning q via the method of the I-LLMP. In this case q factorises into a set of conditional marginals, which allows each LLM expert to focus on only predicting one point. This factorisation of q implies factorisation of $\widetilde{q}$ (due to the diagonal covariance in Eq. 2):

$$q(\mathcal{Y}|\mathcal{X},\mathcal{D}) = \prod_{j=1}^{m} q(\boldsymbol{y}^j|\boldsymbol{x}^j,\mathcal{D}) \implies \widetilde{q}_{r_t}(\mathcal{Y}|\mathcal{X},\mathcal{D}) = \prod_{j=1}^{m} \widetilde{q}_{r_t}(\widehat{\boldsymbol{y}}^j_{1|t}|\boldsymbol{x}^j,\mathcal{D}).$$

We here defined $\widehat{\boldsymbol{y}}^j_{1|t}(\mathcal{Y}') \coloneqq \mathbb{E}[\boldsymbol{y}^j_1|\mathcal{Y}_t = \mathcal{Y}',\mathcal{X},\mathcal{D}]$, *i.e.* the expected value of a particular output via the flow network's denoised (Tweedie) estimate, which depends simultaneously on all the $\boldsymbol{y}_t$ values at the current time (because all target outputs are simultaneously generated, capturing joint dependencies).

From this we derive a score function like that in Eq. (3) for sampling of product-of-expert distribution π_1 via

$$\nabla_{\mathcal{Y}_t} \log \pi_t(\mathcal{Y}_t|\mathcal{X},\mathcal{D}) \approx s_t^\theta(\mathcal{Y}_t;\mathcal{X},\mathcal{D}) + \sum_{j=1}^{m} \nabla_{\mathcal{Y}_t} \log \widetilde{q}_{r_t}(\widehat{\boldsymbol{y}}^j_{1|t}|\boldsymbol{x}^j,\mathcal{D}).$$

To calculate the factorised convolved distributions $\widetilde{q}_{r_t}$, we note that I-LLM outputs probability in bins, so the integral can be reformulated as a sum over disjoint bins $B \in \mathcal{B}, B \subset \mathbb{R}^{\text{out}}$ spanning the space. In the simplest one-dimensional case, q is piecewise constant on a series of intervals $B = [a_B, b_B]$ for $B \in \mathcal{B}$ partitioning $\mathbb{R}$ with density values q_B. The smoothed density is then

$$\widetilde{q}_r(y) = \sum_{B\in\mathcal{B}} q_B \left[\Phi\left(\frac{b_B - y}{r}\right) - \Phi\left(\frac{a_B - y}{r}\right)\right], \tag{4}$$

using Φ, the standard Gaussian cumulative distribution function.

Note that s_t^θ here need only be a score estimate on some affine Gaussian path: it can come from an unconditional NDP trained with either a flow or a diffusion objective, conditioned via guidance as in Section 2.2, or from a directly conditional model such as that of Hamad and Rosenbaum [17]. With their equivariant architectures our implied product-of-experts probability distribution is also equivariant under re-ordering of $(\boldsymbol{x}^j, \boldsymbol{y}^j)$ target pairs. This equivariance in context predictions is not shared by the autoregressive A-LLMP, since predictions will depend on the rollout ordering.

[7] Since $\widetilde{q}_{r_t}(\boldsymbol{y}) = \int q(\boldsymbol{y}')\,\mathcal{N}(\boldsymbol{y}';\,\boldsymbol{y}, r_t^2 I)\,\mathrm{d}\boldsymbol{y}' = \int q(\boldsymbol{y}')\,\mathcal{N}(\boldsymbol{y}' - \boldsymbol{y}; 0, r_t^2 I)\,\mathrm{d}\boldsymbol{y}'$.

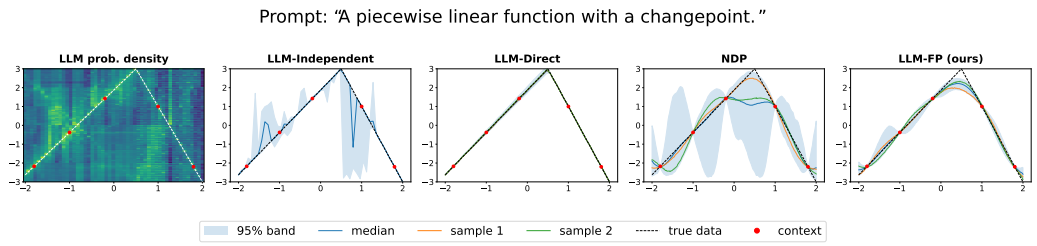

Figure 2: Change-point regression with N=5 context points. Methods with access to the textual change-point description (LLM-Direct, LLM-Independent, LLM-FP) recover the change in slope; the unconditional NDP reverts to its periodic/RBF prior. Two joint samples are shown for methods that produce coherent samples (LLM-Direct, NDP, LLM-FP); the LLM marginal density used by LLM-Independent and LLM-FP is shown on the left.

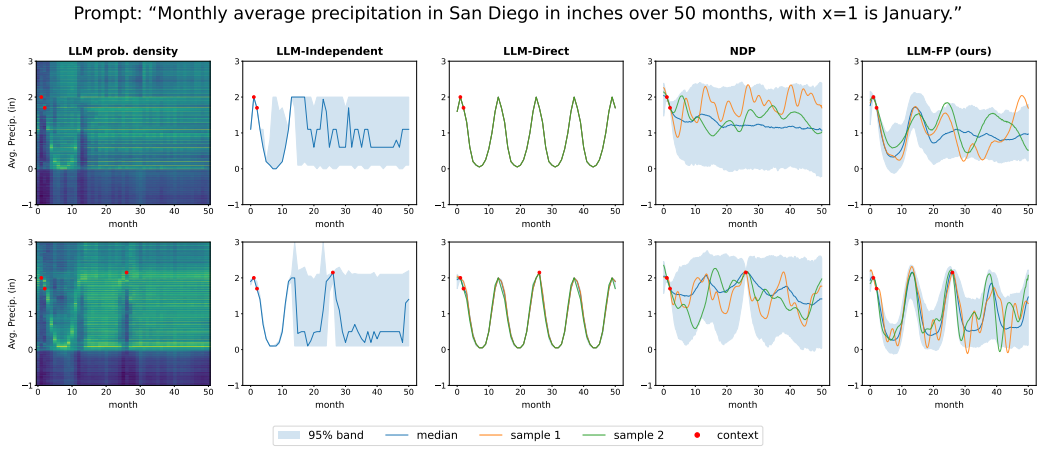

Figure 3: Synthetic San Diego rainfall extrapolation with N=2 (top) and N=3 (bottom) observed values and a shared textual prompt describing seasonal structure. LLM-Direct collapses to a near-deterministic seasonal template in both cases, largely insensitive to the additional observation. The independent LLMP assigns broad marginal mass beyond the first year. The LLM-FP concentrates on periodic joint samples of approximately the correct amplitude and period; adding a third observation further narrows the inferred period and amplitude.

5 Empirical Evaluation

NDP base model. We use a modernised NDP with flow matching objective, DiT backbone and Gaussian conditioning [46] in place of the SDE-based diffusion and in-paint conditioning of Dutordoir et al. [10]; see Appendix D for further details. We found this modernisation, which gave substantial improvements, essential for coherent conditional and unconditional generation (*cf.* for example Fig. 3 in NDP showing samples that clearly do not match their training distribution versus Figs. 6 and 7 in Appendix D.1). We note this improved *NDP* provides a considerably more challenging baseline for comparison than the original model. Like Dutordoir et al. [10] we train on a simple synthetic dataset of Gaussian Process samples of mixed kernel types and length-scales (somewhat richer than in the above, for example, including periodic kernels, see Appendix D for details); the NDP implicitly marginalizes over these and gives a Bayesian posterior prediction based on this data as a prior [40].

Baselines. We compare LLM-FP against three baselines that isolate different design choices in this space: our improved NDP without LLM conditioning (*NDP*); an autoregressive LLM predicting one point at a time (*LLM-Direct*); and independent LLM predictions (*LLM-Independent*).

For the autoregressive *LLM-Direct*, we follow the Direct Prompt constrained generation method of Williams et al. [63]. We also compute independent LLM probabilities using binning, similarly to Requeima et al. [50]. The binned LLM log-probabilities used by LLM-FP are identical to the

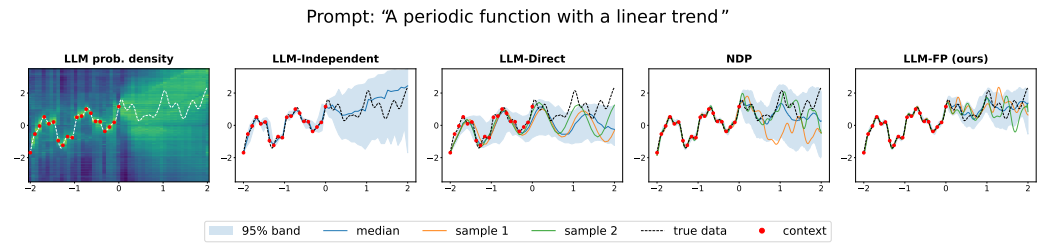

Figure 4: Extrapolation on a periodic function with an added linear trend, with N=20 context points in the range [-2, 0]. The periodic component is consistent with the NDP training prior but the linear trend is not: the NDP remains well-calibrated but reverts towards its prior mean away from the observations. LLM-Direct fails to fit the context and the true function lies outside its predictive distribution. The independent LLMP captures the upward trend but with very broad marginals. LLM-FP follows the trend while retaining narrower uncertainty than the independent LLMP.

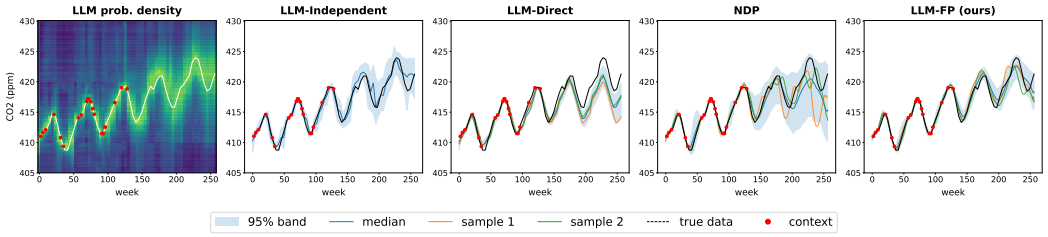

Figure 5: Mauna Loa atmospheric CO_2 measurements, with N=20 randomly selected weeks from 2019-01-06 to 2021-06-13 as context (full prompt in Appendix F). The independent LLMP marginals are approximately correct with broad pointwise uncertainty; NDP joint samples are locally smooth but drift from the true function in phase and amplitude over the window; LLM-Direct is sharply overconfident (NLPD is inflated, see Table 1). LLM-FP combines the two, staying close to the true function throughout.

marginals sampled by *LLM-Independent*; we display these in figures as *LLM log probs*. *LLM-Independent* is sampled directly from these.

All LLM-based methods (LLM-Direct, LLM-Independent, LLM-FP) use the Qwen-3.6-27B LLM [49]. We add a reasoning block before predictions for all LLM-based methods which we detail in Appendix F. These methods use identical prompts within each scenario; prompts are not tuned per method. Note that we did not do any fine-tuning of our LLMs; the fine-tuning of Appendix J applies to the NDP only.

Across scenarios, the autoregressive *LLM-Direct* predicts at 51–64 points as does *LLM-Independent*; this is a favourable regime for *LLM-Direct*, since autoregressive performance degrades severely beyond roughly 100 points (Fig. 1). NDP and LLM-FP predict at 256 points throughout without difficulty; LLM-FP uses LLM evaluations only at the same points as *LLM-Independent*, since the NDP provides the remaining structure, representing a $\sim 4\times$ computational saving over LLM predictions at all 256 points.

Metrics. In all evaluations with a ground truth we include the mean absolute error (MAE), the continuous ranked probability score (CRPS) which captures some degree of calibration, and the negative log likelihood of per-point fitted Gaussians (NLPD), which better captures overconfidence.

Scenarios. We evaluate across four scenarios: a simple change-point function to test text faithfulness (Fig. 2); rainfall extrapolation (Fig. 3) as used in [50]; a synthetic periodic with trend function (Fig. 4); and a real-data problem from the Mauna Loa atmospheric dataset [45] (Fig. 5). Since most of our benchmarks are indexed by a single scalar input such as time, they could also be viewed as timeseries; however, our setting makes no assumption of causal ordering, all predictions condition on the entire context, and indeed the inputs $\boldsymbol{x}$ can be multi-dimensional.

Results. Across the four scenarios (Figs. 2–5), LLM-FP produces joint samples that are simultaneously coherent (unlike LLM-Independent) and faithful to the textual context (unlike NDP), avoid-

Table 1: Predictive performance across regression tasks; lower is better, best per row in bold. Cells are central ± half-width of the 95% bootstrap CI over evaluation positions.

Task	Metric	LLM-Independent	LLM-Direct	NDP	LLM-FP (ours)
Change point	MAE	$0.29_{\pm.13}$	$\mathbf{0.03_{\pm.01}}$	$0.53_{\pm.17}$	$0.13_{\pm.05}$
	CRPS	$0.12_{\pm.06}$	$\mathbf{0.01_{\pm.00}}$	$0.31_{\pm.10}$	$0.10_{\pm.04}$
	NLPD	$0.06_{\pm.27}$	$\mathbf{-1.77_{\pm.30}}$	$0.62_{\pm.26}$	$-0.23_{\pm.63}$
Periodic + linear	MAE	$0.27_{\pm.08}$	$0.57_{\pm.16}$	$0.27_{\pm.11}$	$\mathbf{0.15_{\pm.05}}$
	CRPS	$0.22_{\pm.05}$	$0.42_{\pm.13}$	$0.18_{\pm.07}$	$\mathbf{0.10_{\pm.04}}$
	NLPD	$0.24_{\pm.23}$	$0.89_{\pm.37}$	$-0.39_{\pm.37}$	$\mathbf{-0.73_{\pm.31}}$
Mauna Loa CO_2 (weekly)	MAE	$0.60_{\pm.14}$	$1.05_{\pm.24}$	$0.87_{\pm.29}$	$\mathbf{0.51_{\pm.13}}$
	CRPS	$0.43_{\pm.10}$	$0.81_{\pm.19}$	$0.57_{\pm.18}$	$\mathbf{0.36_{\pm.10}}$
	NLPD	$0.91_{\pm.21}$	$1.95_{\pm.53}$	$0.92_{\pm.30}$	$\mathbf{0.69_{\pm.27}}$

ing both the autoregressive collapse of LLM-Direct and the broad marginals of LLM-Independent. Quantitatively, LLM-FP achieves the best or near-best CRPS and NLPD on Periodic+linear and Mauna Loa [45] (Table 1); on the text-dominated Change point problem, LLM-FP smooths out the occasional outliers in the LLM-Independent samples through the NDP's local consistency, but does not match LLM-Direct's near-perfect reproduction of the change point. The appendix reports three further studies: fine-tuning the NDP on real data (Appendix J), a weaker LLM expert (Appendix K) and a sweep over the number of context points (Appendix L).

6 Discussion

Mechanism. LLM-FP combines two complementary sources of information: the NDP is trained to provide a coherent stochastic process prior, producing smooth and time-consistent joint samples, while the LLM provides semantic information from the textual prompt. The product-of-experts conditioning concentrates probability mass on trajectories that are probable under both factors, with the result that LLM-FP samples are typically more concentrated than either component used in isolation (see, e.g., Figs. 3, 4). This sharpening is the intended effect of combining factors that carry independent information. On the tasks where the LLM marginals are reliable, LLM-FP is both more concentrated than its components and better in both CRPS and NLPD (Table 1). This is most useful when a task requires both sources of information and neither alone suffices. Where the text alone primarily determines the answer the LLM baselines are better (LLM-Direct on Change point, Table 1; both LLM baselines at the smallest context sizes of Appendix L); where the numerical context alone determines it the NDP is (the fine-tuned series of Appendix J). Between these extremes, combining them wins. Additionally the NDP component of the LLM-FP can provide some additional robustness by exploiting local structures; Fig. 2 shows that even the median LLM-Independent prediction can have outlier predictions at certain input points, a problem the LLM-FP removes.

Text as function-class selection. One way to view our LLM conditioning is that it performs a form of soft function-class selection, an interpretation we make precise in Appendix C. In our experiments, the NDP prior marginalises over functions drawn from a mixture of periodic and RBF kernels; when the text describes seasonal or periodic behaviour, the LLM-derived probabilities suppress trajectories inconsistent with this description, effectively reweighting the NDP prior towards its periodic component. In Fig. 4, the LLM-derived probabilities favour upward-trending predictions that are not directly represented by the NDP training prior, allowing the combined model to extrapolate behaviour that lies near the boundary of the NDP support. In both cases, the text does not replace the process prior, but selects among, or extrapolates from, behaviours already assigned non-negligible probability by the NDP. Appendix M quantifies this reweighting on the San Diego task.

Failure modes of direct LLM regression. The autoregressive LLM-Direct is strong on text-dominated problems that suit its linear-trend attractor state (Fig. 2), but fragile once unrolled over many locations: it collapses to a near-deterministic template (Fig. 3) and degrades as the number of prediction points grows (Fig. 1). The independent LLMP avoids this by treating each location separately, at the cost of joint coherence. LLM-FP conditions on those marginals while keeping the NDP's joint structure, sidestepping both failure modes. It is the best method at every context size

from $n = 10$ upwards; at sparser settings, where the NDP has too little data to contribute, the LLM baselines are better (Appendix L).

Limitations. The product-of-experts update implicitly treats the NDP and LLM factors as independent. If the two are strongly correlated, the update may double-count evidence and concentrate further than the evidence warrants; this risk is exacerbated when the LLM-derived probabilities are miscalibrated or overconfident. Over-concentration can be measured as a degradation in CRPS and NLPD. Such degradation does not appear for the 27B LLM used throughout the experiments in the main paper. We speculate that this is because the NDP and LLM are integrating different sources of knowledge for their predictions: the independent LLM integrates textual information at individual points, while the NDP leans on the local function structures contained in its trajectories. Over-concentration can however be seen at very sparse context ($n \leq 5$, Appendix L), where the NDP has too little data to constrain the combined distribution. LLM-FP's NLPD is then worse than that of both of its components, despite a much lower MAE than the NDP. To a degree this over-concentration also appears with a weaker expert on Periodic + linear (Appendix K), where LLM-FP's NLPD is worse than that of the NDP alone, although it still outperforms the other LLM-based methods on every task and metric.

More broadly, the quality of the combined model depends on both the calibration of the LLM marginals and on whether the NDP prior assigns sufficient mass to the relevant functions: if the desired behaviour lies far outside the NDP support, LLM conditioning can only reweigh the existing prior rather than create new trajectory structure (this is one source of the LLM-FP shortfall on Change point in Table 1); conversely, if the LLM marginals are too diffuse, they provide only weak conditioning. This restriction of the LLM-FP to the NDP prior can also be viewed as desirable, as it both allows precise control over the functions we are marginalising over and can filter out erroneous LLM predictions (such as can be seen in the LLM-Independent predictions in Fig. 2). Our main experiments use a single 27B-parameter open-weight LLM without fine-tuning and a simple synthetic NDP training distribution. We study the effects of better or worse models for each component of the LLM-FP: we trial a weaker LLM expert (Appendix K) and fine-tune the NDP on real data (Appendix J) to simulate the effect of a more powerful NDP prior. This also demonstrates the training of the NDP is not restricted to synthetic data, although with the diversity of such methods available (see Section 2.1) we do not see this as a significant limitation. We do not investigate richer NDP priors in depth, nor the effect of prompt design on performance.

7 Conclusion

We introduced LLM-FP, a method for conditioning Neural Diffusion Processes on textual information through LLM-derived probabilities. The approach preserves the coherent trajectory-level samples of NDPs while incorporating semantic information that is unavailable from numerical observations alone. Across synthetic and real regression tasks, this enables text-conditioned predictions that remain smooth and time-consistent, avoiding the instability observed in direct autoregressive LLM regression. The NDP's local trajectory structure also regularises the LLM marginals themselves: pointwise predictions at neighbouring inputs may be individually well-calibrated yet mutually inconsistent, and coupling them through the NDP rules out such inconsistencies.

More broadly, LLM-FPs provide a way to combine synthetic-data-trained stochastic process priors with expert knowledge expressed in language. This suggests a path towards regression models that retain the calibration and joint structure of neural process models while benefiting from the flexible conditioning capabilities of LLMs.

Future Work. Several directions could improve LLM-FP itself. The NDP component would benefit from scaling and from richer synthetic-data generation procedures [23, 24, 64]; linear [26, 8] or neighbourhood [19, 18] attention architectures would reduce the cost of conditioning on large context sets; and text conditioning will naturally improve with stronger LLMs. The product-of-experts construction itself opens further directions. Our gradient-free convolution method applies to any expert distribution that admits a closed-form Gaussian convolution, not only to LLMs with binned predictions. This makes it applicable to guided optimisation in the style of Song et al. [56], Kong et al. [28], quantised measurement models, and potentially to guiding diffusion image models with tokenized language-image experts.

Acknowledgments and Disclosure of Funding

We thank Secondmind and Wayve for supporting this work. The authors acknowledge the use of resources provided by the Isambard-AI National AI Research Resource (AIRR) [39]. Isambard-AI is operated by the University of Bristol and is funded by the UK Government's Department for Business, Innovation, Science and Trade (DBIST) via UK Research and Innovation; and the Science and Technology Facilities Council [ST/AIRR/I-A-I/1023].

References

[1] Michael S. Albergo and Eric Vanden-Eijnden. Building normalizing flows with stochastic interpolants. In *The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023*. OpenReview.net, 2023. URL https://openreview.net/forum?id=li7qeBbCR1t.

[2] Arpit Bansal, Hong-Min Chu, Avi Schwarzschild, Soumyadip Sengupta, Micah Goldblum, Jonas Geiping, and Tom Goldstein. Universal guidance for diffusion models. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net, 2024. URL https://openreview.net/forum?id=pzpWBbnwiJ.

[3] Samy Bengio, Oriol Vinyals, Navdeep Jaitly, and Noam Shazeer. Scheduled sampling for sequence prediction with recurrent neural networks. In Corinna Cortes, Neil D. Lawrence, Daniel D. Lee, Masashi Sugiyama, and Roman Garnett, editors, *Advances in Neural Information Processing Systems 28: Annual Conference on Neural Information Processing Systems 2015, December 7-12, 2015, Montreal, Quebec, Canada*, pages 1171–1179, 2015. URL https://proceedings.neurips.cc/paper/2015/hash/e995f98d56967d946471af29d7bf99f1-Abstract.html.

[4] Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. Language models are few-shot learners. In Hugo Larochelle, Marc'Aurelio Ranzato, Raia Hadsell, Maria-Florina Balcan, and Hsuan-Tien Lin, editors, *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*, 2020. URL https://proceedings.neurips.cc/paper/2020/hash/1457c0d6bfcb4967418bfb8ac142f64a-Abstract.html.

[5] Wessel P. Bruinsma, James Requeima, Andrew Y. K. Foong, Jonathan Gordon, and Richard E. Turner. The gaussian neural process. *CoRR*, abs/2101.03606, 2021. URL https://arxiv.org/abs/2101.03606.

[6] Tian Qi Chen, Yulia Rubanova, Jesse Bettencourt, and David Duvenaud. Neural ordinary differential equations. In Samy Bengio, Hanna M. Wallach, Hugo Larochelle, Kristen Grauman, Nicolò Cesa-Bianchi, and Roman Garnett, editors, *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, December 3-8, 2018, Montréal, Canada*, pages 6572–6583, 2018. URL https://proceedings.neurips.cc/paper/2018/hash/69386f6bb1dfed68692a24c8686939b9-Abstract.html.

[7] Kristy Choi, Chris Cundy, Sanjari Srivastava, and Stefano Ermon. Lmpriors: Pre-trained language models as task-specific priors. *CoRR*, abs/2210.12530, 2022. doi: 10.48550/ARXIV.2210.12530. URL https://doi.org/10.48550/arXiv.2210.12530.

[8] Krzysztof Marcin Choromanski, Valerii Likhosherstov, David Dohan, Xingyou Song, Andreea Gane, Tamás Sarlós, Peter Hawkins, Jared Quincy Davis, Afroz Mohiuddin, Lukasz Kaiser, David Benjamin Belanger, Lucy J. Colwell, and Adrian Weller. Rethinking attention with performers. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net, 2021. URL https://openreview.net/forum?id=Ua6zuk0WRH.

[9] Hyungjin Chung, Jeongsol Kim, Michael Thompson McCann, Marc Louis Klasky, and Jong Chul Ye. Diffusion posterior sampling for general noisy inverse problems. In *The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023*. OpenReview.net, 2023. URL https://openreview.net/forum?id=OnD9zGAGT0k.

[10] Vincent Dutordoir, Alan Saul, Zoubin Ghahramani, and Fergus Simpson. Neural diffusion processes. In Andreas Krause, Emma Brunskill, Kyunghyun Cho, Barbara Engelhardt, Sivan Sabato, and Jonathan Scarlett, editors, *International Conference on Machine Learning, ICML 2023, 23-29 July 2023, Honolulu, Hawaii, USA*, volume 202 of *Proceedings of Machine Learning Research*, pages 8990–9012. PMLR, 2023. URL https://proceedings.mlr.press/v202/dutordoir23a.html.

[11] Marta Garnelo, Dan Rosenbaum, Christopher Maddison, Tiago Ramalho, David Saxton, Murray Shanahan, Yee Whye Teh, Danilo Jimenez Rezende, and S. M. Ali Eslami. Conditional neural processes. In Jennifer G. Dy and Andreas Krause, editors, *Proceedings of the 35th International Conference on Machine Learning, ICML 2018, Stockholmsmässan, Stockholm, Sweden, July 10-15, 2018*, volume 80 of *Proceedings of Machine Learning Research*, pages 1690–1699. PMLR, 2018. URL http://proceedings.mlr.press/v80/garnelo18a.html.

[12] Marta Garnelo, Jonathan Schwarz, Dan Rosenbaum, Fabio Viola, Danilo J. Rezende, S. M. Ali Eslami, and Yee Whye Teh. Neural processes. *CoRR*, abs/1807.01622, 2018. URL http://arxiv.org/abs/1807.01622.

[13] Tilmann Gneiting and Adrian E Raftery. Strictly proper scoring rules, prediction, and estimation. *Journal of the American statistical Association*, 102(477):359–378, 2007.

[14] Will Grathwohl, Ricky T. Q. Chen, Jesse Bettencourt, Ilya Sutskever, and David Duvenaud. FFJORD: free-form continuous dynamics for scalable reversible generative models. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net, 2019. URL https://openreview.net/forum?id=rJxgknCcK7.

[15] Nate Gruver, Marc Finzi, Shikai Qiu, and Andrew Gordon Wilson. Large language models are zero-shot time series forecasters. In Alice Oh, Tristan Naumann, Amir Globerson, Kate Saenko, Moritz Hardt, and Sergey Levine, editors, *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023, New Orleans, LA, USA, December 10 - 16, 2023*, 2023. URL http://papers.nips.cc/paper_files/paper/2023/hash/3eb7ca52e8207697361b2c0fb3926511-Abstract-Conference.html.

[16] Yingqing Guo, Hui Yuan, Yukang Yang, Minshuo Chen, and Mengdi Wang. Gradient guidance for diffusion models: An optimization perspective. In Amir Globersons, Lester Mackey, Danielle Belgrave, Angela Fan, Ulrich Paquet, Jakub M. Tomczak, and Cheng Zhang, editors, *Advances in Neural Information Processing Systems 37: Annual Conference on Neural Information Processing Systems 2024, NeurIPS 2024, Vancouver, BC, Canada, December 10 - 15, 2024*, 2024. URL http://papers.nips.cc/paper_files/paper/2024/hash/a5059a9a389ccc76da85760ea79490d8-Abstract-Conference.html.

[17] Hussen Abu Hamad and Dan Rosenbaum. Flow matching neural processes. In *Advances in Neural Information Processing Systems 38 (NeurIPS 2025)*, 2025. URL https://openreview.net/forum?id=ATPyo6tKXr.

[18] Ali Hassani and Humphrey Shi. Dilated neighborhood attention transformer. *CoRR*, abs/2209.15001, 2022. doi: 10.48550/ARXIV.2209.15001. URL https://doi.org/10.48550/arXiv.2209.15001.

[19] Ali Hassani, Steven Walton, Jiachen Li, Shen Li, and Humphrey Shi. Neighborhood attention transformer. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2023, Vancouver, BC, Canada, June 17-24, 2023*, pages 6185–6194. IEEE, 2023. doi: 10.1109/CVPR52729.2023.00599. URL https://doi.org/10.1109/CVPR52729.2023.00599.

[20] Dan Hendrycks and Kevin Gimpel. Bridging nonlinearities and stochastic regularizers with gaussian error linear units. *CoRR*, abs/1606.08415, 2016. URL http://arxiv.org/abs/1606.08415.

[21] Geoffrey E. Hinton. Training products of experts by minimizing contrastive divergence. *Neural Comput.*, 14(8):1771–1800, 2002. doi: 10.1162/089976602760128018. URL https://doi.org/10.1162/089976602760128018.

[22] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. In Hugo Larochelle, Marc'Aurelio Ranzato, Raia Hadsell, Maria-Florina Balcan, and Hsuan-Tien Lin, editors, *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*, 2020. URL https://proceedings.neurips.cc/paper/2020/hash/4c5bcfec8584af0d967f1ab10179ca4b-Abstract.html.

[23] Noah Hollmann, Samuel Müller, Katharina Eggensperger, and Frank Hutter. Tabpfn: A transformer that solves small tabular classification problems in a second. In *The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023*. OpenReview.net, 2023. URL https://openreview.net/forum?id=cp5PvcI6w8_.

[24] Noah Hollmann, Samuel Müller, Lennart Purucker, Arjun Krishnakumar, Max Körfer, Shi Bin Hoo, Robin Tibor Schirrmeister, and Frank Hutter. Accurate predictions on small data with a tabular foundation model. *Nat.*, 637(8044):319–326, 2025. doi: 10.1038/S41586-024-08328-6. URL https://doi.org/10.1038/s41586-024-08328-6.

[25] Ari Holtzman, Jan Buys, Li Du, Maxwell Forbes, and Yejin Choi. The curious case of neural text degeneration. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net, 2020. URL https://openreview.net/forum?id=rygGQyrFvH.

[26] Angelos Katharopoulos, Apoorv Vyas, Nikolaos Pappas, and François Fleuret. Transformers are rnns: Fast autoregressive transformers with linear attention. In *Proceedings of the 37th International Conference on Machine Learning, ICML 2020, 13-18 July 2020, Virtual Event*, Proceedings of Machine Learning Research, pages 5156–5165. PMLR, 2020. URL http://proceedings.mlr.press/v119/katharopoulos20a.html.

[27] Marcel Kollovieh, Marten Lienen, David Lüdke, Leo Schwinn, and Stephan Günnemann. Flow matching with gaussian process priors for probabilistic time series forecasting. In *The Thirteenth International Conference on Learning Representations, ICLR 2025, Singapore, April 24-28, 2025*. OpenReview.net, 2025. URL https://openreview.net/forum?id=uxVBbSlKQ4.

[28] Lingkai Kong, Yuanqi Du, Wenhao Mu, Kirill Neklyudov, Valentin De Bortoli, Dongxia Wu, Haorui Wang, Aaron M. Ferber, Yian Ma, Carla P. Gomes, and Chao Zhang. Diffusion models as constrained samplers for optimization with unknown constraints. In Yingzhen Li, Stephan Mandt, Shipra Agrawal, and Mohammad Emtiyaz Khan, editors, *International Conference on Artificial Intelligence and Statistics, AISTATS 2025, Mai Khao, Thailand, 3-5 May 2025*, volume 258 of *Proceedings of Machine Learning Research*, pages 4582–4590. PMLR, 2025. URL https://proceedings.mlr.press/v258/kong25b.html.

[29] X. Lan, G. Pétron, K. Baugh, A. M. Crotwell, M. J. Crotwell, S. DeVogel, M. Madronich, J. Mauss, T. Mefford, E. Moglia, S. Morris, J. W. Mund, A. Searle, and J. Miller. Atmospheric methane dry air mole fractions from the NOAA GML global greenhouse gas reference network, carbon cycle cooperative global air sampling network: 1983–present, 2026. Version 2026-07-17.

[30] X. Lan, G. Pétron, K. Baugh, A. M. Crotwell, M. J. Crotwell, S. DeVogel, M. Madronich, J. Mauss, T. Mefford, E. Moglia, S. Morris, J. W. Mund, A. Searle, and J. Miller. Atmospheric nitrous oxide dry air mole fractions from the NOAA GML global greenhouse gas reference network, carbon cycle cooperative global air sampling network: 1997–present, 2026. Version 2026-07-17.

[31] X. Lan, G. Pétron, K. Baugh, A. M. Crotwell, M. J. Crotwell, S. DeVogel, M. Madronich, J. Mauss, T. Mefford, E. Moglia, S. Morris, J. W. Mund, A. Searle, and J. Miller. Atmospheric sulfur hexafluoride dry air mole fractions from the NOAA GML global greenhouse gas reference network, carbon cycle cooperative global air sampling network: 1997–present, 2026. Version 2026-07-17.

[32] Yann LeCun. Why autoregressive language models are exponentially diverging and doomed. LinkedIn post, May 2023.

[33] Yaron Lipman, Ricky T. Q. Chen, Heli Ben-Hamu, Maximilian Nickel, and Matthew Le. Flow matching for generative modeling. In *The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023*. OpenReview.net, 2023. URL https://openreview.net/forum?id=PqvMRDCJT9t.

[34] Yaron Lipman, Marton Havasi, Peter Holderrieth, Neta Shaul, Matt Le, Brian Karrer, Ricky T. Q. Chen, David Lopez-Paz, Heli Ben-Hamu, and Itai Gat. Flow matching guide and code. *CoRR*, abs/2412.06264, 2024. doi: 10.48550/ARXIV.2412.06264. URL https://doi.org/10.48550/arXiv.2412.06264.

[35] Xingchao Liu, Chengyue Gong, and Qiang Liu. Flow straight and fast: Learning to generate and transfer data with rectified flow. In *The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023*. OpenReview.net, 2023. URL https://openreview.net/forum?id=XVjTT1nw5z.

[36] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net, 2019. URL https://openreview.net/forum?id=Bkg6RiCqY7.

[37] Andreas Lugmayr, Martin Danelljan, Andrés Romero, Fisher Yu, Radu Timofte, and Luc Van Gool. Repaint: Inpainting using denoising diffusion probabilistic models. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2022, New Orleans, LA, USA, June 18-24, 2022*, pages 11451–11461. IEEE, 2022. doi: 10.1109/CVPR52688.2022.01117. URL https://doi.org/10.1109/CVPR52688.2022.01117.

[38] Stratis Markou, James Requeima, Wessel P. Bruinsma, and Richard E. Turner. Efficient gaussian neural processes for regression. *CoRR*, abs/2108.09676, 2021. URL https://arxiv.org/abs/2108.09676.

[39] Simon McIntosh-Smith, Sadaf R. Alam, and Christopher Woods. Isambard-AI: a leadership class supercomputer optimised specifically for Artificial Intelligence, 2024. URL https://doi.org/10.48550/arXiv.2410.11199.

[40] Samuel Müller, Noah Hollmann, Sebastian Pineda-Arango, Josif Grabocka, and Frank Hutter. Transformers can do bayesian inference. In *The Tenth International Conference on Learning Representations, ICLR 2022, Virtual Event, April 25-29, 2022*. OpenReview.net, 2022. URL https://openreview.net/forum?id=KSugKcbNf9.

[41] Samuel Müller, Arik Reuter, Noah Hollmann, David Rügamer, and Frank Hutter. Position: The future of bayesian prediction is prior-fitted. In *Forty-second International Conference on Machine Learning, ICML 2025, Vancouver, BC, Canada, July 13-19, 2025 - Position Paper Track*. OpenReview.net, 2025. URL https://proceedings.mlr.press/v267/muller25d.html.

[42] Judea Pearl. *Probabilistic reasoning in intelligent systems - networks of plausible inference*. Morgan Kaufmann series in representation and reasoning. Morgan Kaufmann, 1989.

[43] William Peebles and Saining Xie. Scalable diffusion models with transformers. In *IEEE/CVF International Conference on Computer Vision, ICCV 2023, Paris, France, October 1-6, 2023*, pages 4172–4182. IEEE, 2023. doi: 10.1109/ICCV51070.2023.00387. URL https://doi.org/10.1109/ICCV51070.2023.00387.

[44] Matthew Peroni, Franck Le, and Vadim Sheinin. Robust tabular foundation models. *CoRR*, abs/2512.03307, 2025. doi: 10.48550/ARXIV.2512.03307. URL https://doi.org/10.48550/arXiv.2512.03307.

[45] G. Pétron, P. Handley, J. Hakala, J. Kofler, K. W. Thoning, A. Crotwell, A. Clarke, J. Fernandez, I. Crocker, R. Burgener, B. Thomas, A. Colton, S. Jordan, L. Longren, M. Doneghey, M. Trudeau, T. Mefford, S. DeVogel, A. Searle, X. Lan, J. Mauss, C. Smith, B. Vasel, S. Morris, J. Miller, and J. Mund. Atmospheric carbon dioxide dry air mole fractions from continuous measurements at mauna loa, hawaii, utqiaġvik, alaska, american samoa and south pole, 1973–present, 2026. Version 2026-08-11.

[46] Ashwini Pokle, Matthew J. Muckley, Ricky T. Q. Chen, and Brian Karrer. Training-free linear image inverses via flows. *Trans. Mach. Learn. Res.*, 2024, 2024. URL https://openreview.net/forum?id=PLIt3a4yTm.

[47] Jingang Qu, David Holzmüller, Gaël Varoquaux, and Marine Le Morvan. Tabiclv2: A better, faster, scalable, and open tabular foundation model. *CoRR*, abs/2602.11139, 2026. doi: 10.48550/ARXIV.2602.11139. URL https://doi.org/10.48550/arXiv.2602.11139.

[48] Qwen Team. Qwen3.5: Towards native multimodal agents, February 2026. URL https://qwen.ai/blog?id=qwen3.5.

[49] Qwen Team. Qwen3.6-27B: Flagship-level coding in a 27b dense model, April 2026. URL https://qwen.ai/blog?id=qwen3.6-27b.

[50] James Requeima, John Bronskill, Dami Choi, Richard E. Turner, and David Kristjanson Duvenaud. LLM processes: Numerical predictive distributions conditioned on natural language. In Amir Globersons, Lester Mackey, Danielle Belgrave, Angela Fan, Ulrich Paquet, Jakub M. Tomczak, and Cheng Zhang, editors, *Advances in Neural Information Processing Systems 37: Annual Conference on Neural Information Processing Systems 2024, NeurIPS 2024, Vancouver, BC, Canada, December 10 - 15, 2024*, 2024. URL http://papers.nips.cc/paper_files/paper/2024/hash/c5ec22711f3a4a2f4a0a8ffd92167190-Abstract-Conference.html.

[51] François Rozet and Gilles Louppe. Score-based data assimilation. In Alice Oh, Tristan Naumann, Amir Globerson, Kate Saenko, Moritz Hardt, and Sergey Levine, editors, *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023, New Orleans, LA, USA, December 10 - 16, 2023*, 2023. URL http://papers.nips.cc/paper_files/paper/2023/hash/7f7fa581cc8a1970a4332920cdf87395-Abstract-Conference.html.

[52] François Rozet, Gérôme Andry, François Lanusse, and Gilles Louppe. Learning diffusion priors from observations by expectation maximization. In Amir Globersons, Lester Mackey, Danielle Belgrave, Angela Fan, Ulrich Paquet, Jakub M. Tomczak, and Cheng Zhang, editors, *Advances in Neural Information Processing Systems 37: Annual Conference on Neural Information Processing Systems 2024, NeurIPS 2024, Vancouver, BC, Canada, December 10 - 15, 2024*, 2024. URL http://papers.nips.cc/paper_files/paper/2024/hash/9f94298bac4668db4dc77ddb0a244301-Abstract-Conference.html.

[53] Yifei Shen, Xinyang Jiang, Yifan Yang, Yezhen Wang, Dongqi Han, and Dongsheng Li. Understanding and improving training-free loss-based diffusion guidance. In Amir Globersons, Lester Mackey, Danielle Belgrave, Angela Fan, Ulrich Paquet, Jakub M. Tomczak, and Cheng Zhang, editors, *Advances in Neural Information Processing Systems 37: Annual Conference on Neural Information Processing Systems 2024, NeurIPS 2024, Vancouver, BC, Canada, December 10 - 15, 2024*, 2024. URL http://papers.nips.cc/paper_files/paper/2024/hash/c4edc5113b4ffd4632718558fb66b9ef-Abstract-Conference.html.

[54] Jascha Sohl-Dickstein, Eric A. Weiss, Niru Maheswaranathan, and Surya Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics. In Francis R. Bach and David M. Blei, editors, *Proceedings of the 32nd International Conference on Machine Learning, ICML 2015, Lille, France, 6-11 July 2015*, volume 37 of *JMLR Workshop and Conference Proceedings*, pages 2256–2265. JMLR.org, 2015. URL http://proceedings.mlr.press/v37/sohl-dickstein15.html.

[55] Jiaming Song, Arash Vahdat, Morteza Mardani, and Jan Kautz. Pseudoinverse-guided diffusion models for inverse problems. In *The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023*. OpenReview.net, 2023. URL https://openreview.net/forum?id=9_gsMA8MRKQ.

[56] Jiaming Song, Qinsheng Zhang, Hongxu Yin, Morteza Mardani, Ming-Yu Liu, Jan Kautz, Yongxin Chen, and Arash Vahdat. Loss-guided diffusion models for plug-and-play controllable generation. In Andreas Krause, Emma Brunskill, Kyunghyun Cho, Barbara Engelhardt, Sivan Sabato, and Jonathan Scarlett, editors, *International Conference on Machine Learning, ICML 2023, 23-29 July 2023, Honolulu, Hawaii, USA*, volume 202 of *Proceedings of Machine Learning Research*, pages 32483–32498. PMLR, 2023. URL https://proceedings.mlr.press/v202/song23k.html.

[57] Yang Song, Jascha Sohl-Dickstein, Diederik P. Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net, 2021. URL https://openreview.net/forum?id=PxTIG12RRHS.

[58] Michael Unser. Splines: a perfect fit for signal and image processing. *IEEE Signal Process. Mag.*, 16(6): 22–38, 1999. doi: 10.1109/79.799930. URL https://doi.org/10.1109/79.799930.

[59] Robert Vacareanu, Vlad-Andrei Negru, Vasile Suciu, and Mihai Surdeanu. From words to numbers: Your large language model is secretly a capable regressor when given in-context examples. *CoRR*, abs/2404.07544, 2024. doi: 10.48550/ARXIV.2404.07544. URL https://doi.org/10.48550/arXiv.2404.07544.

[60] Pascal Vincent. A connection between score matching and denoising autoencoders. *Neural Comput.*, 23 (7):1661–1674, 2011. doi: 10.1162/NECO_A_00142. URL https://doi.org/10.1162/NECO_a_00142.

[61] Zhilin Wang, Yafu Li, Jianhao Yan, Yu Cheng, and Yue Zhang. Unveiling attractor cycles in large language models: A dynamical systems view of successive paraphrasing. In Wanxiang Che, Joyce Nabende, Ekaterina Shutova, and Mohammad Taher Pilehvar, editors, *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2025, Vienna, Austria, July 27 - August 1, 2025*, pages 12740–12755. Association for Computational Linguistics, 2025. URL https://aclanthology.org/2025.acl-long.624/.

[62] Sean Welleck, Ilia Kulikov, Stephen Roller, Emily Dinan, Kyunghyun Cho, and Jason Weston. Neural text generation with unlikelihood training. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net, 2020. URL https://openreview.net/forum?id=SJeYe0NtvH.

[63] Andrew Robert Williams, Arjun Ashok, Étienne Marcotte, Valentina Zantedeschi, Jithendaraa Subramanian, Roland Riachi, James Requeima, Alexandre Lacoste, Irina Rish, Nicolas Chapados, and Alexandre Drouin. Context is key: A benchmark for forecasting with essential textual information. In Aarti Singh, Maryam Fazel, Daniel Hsu, Simon Lacoste-Julien, Felix Berkenkamp, Tegan Maharaj, Kiri Wagstaff, and Jerry Zhu, editors, *Forty-second International Conference on Machine Learning, ICML 2025, Vancouver, BC, Canada, July 13-19, 2025*, Proceedings of Machine Learning Research. PMLR / OpenReview.net, 2025. URL https://proceedings.mlr.press/v267/williams25a.html.

[64] Yulun Wu and Doron L. Bergman. Zero-shot meta-learning for tabular prediction tasks with adversarially pre-trained transformer. *CoRR*, abs/2502.04573, 2025. doi: 10.48550/ARXIV.2502.04573. URL https://doi.org/10.48550/arXiv.2502.04573.

[65] Jin Xu, Xiaojiang Liu, Jianhao Yan, Deng Cai, Huayang Li, and Jian Li. Learning to break the loop: Analyzing and mitigating repetitions for neural text generation. In Sanmi Koyejo, S. Mohamed, A. Agarwal, Danielle Belgrave, K. Cho, and A. Oh, editors, *Advances in Neural Information Processing Systems 35: Annual Conference on Neural Information Processing Systems 2022, NeurIPS 2022, New Orleans, LA, USA, November 28 - December 9, 2022*, 2022. URL http://papers.nips.cc/paper_files/paper/2022/hash/148c0aeea1c5da82f4fa86a09d4190da-Abstract-Conference.html.

[66] Han-Jia Ye, Si-Yang Liu, and Wei-Lun Chao. A closer look at TabPFN v2: Strength, limitation, and extension. *CoRR*, abs/2502.17361, 2025. doi: 10.48550/ARXIV.2502.17361. URL https://doi.org/10.48550/arXiv.2502.17361.

[67] Jiwen Yu, Yinhuai Wang, Chen Zhao, Bernard Ghanem, and Jian Zhang. Freedom: Training-free energy-guided conditional diffusion model. In *IEEE/CVF International Conference on Computer Vision, ICCV 2023, Paris, France, October 1-6, 2023*, pages 23117–23127. IEEE, 2023. doi: 10.1109/ICCV51070.2023.02118. URL https://doi.org/10.1109/ICCV51070.2023.02118.

[68] Xiyuan Zhang, Danielle C. Maddix, Junming Yin, Nick Erickson, Abdul Fatir Ansari, Boran Han, Shuai Zhang, Leman Akoglu, Christos Faloutsos, Michael W. Mahoney, Cuixiong Hu, Huzefa Rangwala, George Karypis, and Bernie Wang. Mitra: Mixed synthetic priors for enhancing tabular foundation models. *CoRR*, abs/2510.21204, 2025. doi: 10.48550/ARXIV.2510.21204. URL https://doi.org/10.48550/arXiv.2510.21204.

A Stochastic Processes and Our Models

Suppose we have a distribution $\rho_{x_1,\ldots,x_n}(y_1,\ldots,y_n)$ indexed and defined for any locations $x_1,\ldots,x_n$ and $n \geq 1$. The Kolmogorov Extension Theorem gives sufficient conditions for defining a stochastic process from these as finite dimensional marginals. The conditions are

1. *Exchangeability*: For any permutation π of integers $\{1,\ldots,\mathrm{n}\}$,
$$\rho_{x_1,\ldots,x_n}(y_1,\ldots,y_n) = \rho_{x_{\pi(1)},\ldots,x_{\pi(n)}}(y_{\pi(1)},\ldots,y_{\pi(n)}).$$
2. *Consistency*: For any $1 \leq m \leq n$,
$$\rho_{x_1,\ldots,x_m}(y_1,\ldots,y_m) = \int \rho_{x_1,\ldots,x_n}(y_1,\ldots,y_n)\mathrm{d}y_{m+1}\ldots\mathrm{d}y_n.$$

An NDP defined by a flow model, the I-LLMP as defined in Requeima et al. [50], and therefore our LLM-FP model all satisfy the *exchangeability* property, while the autoregressive A-LLMP does not. They are all unlikely to satisfy the *consistency* property. However the NDP could *learn* to be consistent: if it is trained on datasets satisfying consistency, for example Gaussian Process samples, in the limit of perfect learning it exactly models the Bayesian posterior which is consistent.

B Flow Matching and Diffusion Additional Details

Flow matching (1, 33, 35; see 34 for a comprehensive introduction) generalises the training method for diffusion models [54, 22, 57], framing them as a simulation-free training method for continuous normalising flows [6, 14]. It defines a flow with velocity field $u_t = \frac{\mathrm{d}\boldsymbol{y}_t}{\mathrm{d}t}$ giving rise to a Markov process $(\boldsymbol{Y}_t \sim p_t)_{0\leq t\leq 1}$ from noise $p_{\text{noise}} = p_0$ to data $p_{\text{data}} = p_1$. We would like to learn a parameterised approximation to the velocity field $u_t^\theta \approx u_t$. Sampling could then be straightforwardly performed by integration of the ODE defined via this learned velocity field.

The core result of flow matching is that we can replace the ideal flow matching objective $\mathcal{L}_{FM} = \mathbb{E}_{t,\boldsymbol{Y}_t\sim p_t}\|u_t(\boldsymbol{Y}_t) - u_t^\theta(\boldsymbol{Y}_t)\|^2$ with a version using conditional flows and obtain the same gradients with respect to θ. For example, in the 'independent coupling' formulation, we can introduce conditional probability $p_{t|1}(\boldsymbol{Y}_t|\boldsymbol{Y}_1)$ where integration over $\boldsymbol{Y}_1$ gives the marginals p_t.

From these, we obtain the conditional velocity fields $u_{t|1}(\cdot|\boldsymbol{y}_1)$ which generate the conditional probabilities, and the conditional flow matching objective $\mathcal{L}_{CFM} = \mathbb{E}_{t,\boldsymbol{Y}_1\sim p_1,\boldsymbol{Y}_t\sim p_{t|1}(\cdot|\boldsymbol{Y}_1)}\|u_{t|1}(\boldsymbol{Y}_t|\boldsymbol{Y}_1) - u_t^\theta(\boldsymbol{Y}_t)\|^2$.

Here we exclusively use the popular class (relating closely to diffusion models) of affine conditional flows $\boldsymbol{Y}_t = \alpha_t\boldsymbol{Y}_1 + \sigma_t\boldsymbol{Y}_0$, with a Gaussian noise p_0 distribution. Here the pair (α_t, σ_t) is a 'scheduler'; these are smooth functions with $\alpha_0 = \sigma_1 = 0, \alpha_1 = \sigma_0 = 1$, and $\dot\alpha_t, -\dot\sigma_t > 0$ on $t \in (0,1)$. This gives conditional probability paths
$$p_{t|1}(\boldsymbol{y}|\boldsymbol{y}_1) = \mathcal{N}(\boldsymbol{y}|\alpha_t\boldsymbol{y}_1, \sigma_t^2 I)$$
and flow matching objective
$$\mathbb{E}_{t,\boldsymbol{Y}_1\sim p_1,\boldsymbol{Y}_t\sim p_{t|1}(\cdot|\boldsymbol{Y}_1)}\|\dot\alpha_t\boldsymbol{Y}_1 + \dot\sigma_t\boldsymbol{Y}_0 - u_t^\theta(\boldsymbol{Y}_t)\|^2.$$

As discussed in Section 2, we use the conditional Optimal Transport (OT) path [33]: $\alpha_t = t$ and $\sigma_t = 1 - t$.

For this special class, due to the Gaussian form of $p_{t|1}$, the conditional score function can be written as $\nabla_{\boldsymbol{y}}\log p_{t|1}(\boldsymbol{y}|\boldsymbol{y}_1) = -\frac{1}{\sigma_t^2}(\boldsymbol{y} - \alpha_t\boldsymbol{y}_1)$. Through this, the loss function can be further re-parametrised to frame it as learning an approximation to the marginal score functions, $\nabla_{\boldsymbol{y}_t}\log p_t(\boldsymbol{y}_t)$, reproducing the de-noising score matching objective [60]. Other equivalent losses and rescaled variations can also be used, for example predicting the expected final output directly, $\widehat{\boldsymbol{y}}_{1|t}(\boldsymbol{y}_t) \coloneqq \mathbb{E}[\boldsymbol{Y}_1|\boldsymbol{Y}_t = \boldsymbol{y}_t]$. These can be linearly converted to the flow formulation via[8]

$$u_t(\boldsymbol{y}) = a_t\boldsymbol{y} + b_t\nabla_{\boldsymbol{y}}\log p_t(\boldsymbol{y}) \tag{5}$$
$$\widehat{\boldsymbol{y}}_{1|t}(\boldsymbol{y}) = c_t\boldsymbol{y} + d_t u_t(\boldsymbol{y}). \tag{6}$$

For OT flow $u_t(\boldsymbol{y}) = \frac{1}{t}\boldsymbol{y} + \frac{1-t}{t}\nabla_{\boldsymbol{y}}\log p_t(\boldsymbol{y})$ and $\widehat{\boldsymbol{y}}_{1|t}(\boldsymbol{y}) = \boldsymbol{y} + (1-t)u_t(\boldsymbol{y})$; this straight path is particularly amenable to discretisation when sampling, motivating the choice.

[8] Where we have defined parameters

$$a_t \coloneqq \frac{\dot\alpha_t}{\alpha_t}, \qquad b_t \coloneqq \frac{\sigma_t^2\dot\alpha_t - \sigma_t\dot\sigma_t\alpha_t}{\alpha_t}, \qquad c_t \coloneqq \frac{\dot\sigma_t}{\dot\sigma_t\alpha_t - \sigma_t\dot\alpha_t}, \qquad d_t \coloneqq \frac{\sigma_t}{\sigma_t\dot\alpha_t - \dot\sigma_t\alpha_t}$$

derived through manipulations of the affine path (see *e.g.* Lipman et al. [34]).

Equation (5) makes the relationship between flows and learned score functions clear, the resulting ODE exactly matching with the probability flow ODE seen in diffusion models. To avoid the singularity in Eq. (5) as $t \to 0$, we can begin at $0 < t \ll 1$ where p_t approximates a Gaussian.

We note that by rearranging Eq. (5), u_t^θ can be used in the above to define an approximation to the score function at each timestep, s_t^θ. This can additionally be used for Langevin steps at any fixed t (predictor-corrector; 57), giving *stochastic sampling* of the learned model; combining a Langevin step with every step of the ODE gives the "reverse" stochastic differential equation formulation used by many diffusion models. Although in principle we could sample directly from the score at the final time step (*i.e.* Langevin sampling on p_1), this is in practice infeasible, as it is almost impossible to initialise the sampling process in the support of the model (*i.e.* where it has seen training examples), which would lead to poor samples. This is exactly the problem that flow and diffusion sampling solve, starting at p_0 or p_t with $0 < t \ll 1$ the support of the model is close to Gaussian for which it is trivial to draw initialisations.

B.1 Relationship of flow matching to diffusion.

Diffusion uses a noising process from data constructed from an stochastic differential equation (SDE) with affine drift coefficients and Brownian motion. Various motivations lead to objectives like the de-noising score matching loss $\mathcal{L}_{\text{SM}}$. This happens because we choose an SDE which leads to Gaussian paths $p_{t|1}$, and thus diffusion can be seen as a special case of conditional flow matching, with different exact SDE formulations leading to different schedulers. The OT flow is particularly straightforward to learn as it leads to constant velocity flow. In addition, we note that in the diffusion literature, the parametrisation of time is usually instead defined from data at 0 to noise as $t \to \infty$.

To be specific, diffusion models choose an SDE with affine drift coefficients, and Brownian motion $\boldsymbol{W}_r$,

$$\mathrm{d}\widetilde{\boldsymbol{X}}_r = a_r \widetilde{\boldsymbol{X}}_r \,\mathrm{d}r + g_r \mathrm{d}\boldsymbol{W}_r, \quad \widetilde{\boldsymbol{X}}_0 \sim p_{\text{data}} \quad r \in [0, \infty)$$

and define $\boldsymbol{X}_{k^{-1}(r)} = \widetilde{\boldsymbol{X}}_r$ for some mapping $k : (0, 1] \to [0, \infty)$; this should be strictly monotonically decreasing with $k(1) = 0$ and $\lim_{t\to 0} k(t) = \infty$. Since $\boldsymbol{X}_0$ is Gaussian, we can define the "forward" (using the reversed diffusion time convention) process from data to time t, in closed form as $p_{t|1}(\boldsymbol{X}_t | \boldsymbol{X}_1 = \boldsymbol{x}) = \mathcal{N}(\alpha_t \boldsymbol{x}, \sigma_t^2 I)$, using coefficients α_t and σ_t derived from a_r and g_r (specifically, as $\alpha_t = \tilde{\alpha}_r, \sigma_t = \tilde{\sigma}_r$ for $\tilde{\alpha}_r = \exp \int_0^r a_s \mathrm{d}s$ and $\tilde{\sigma}_r^2 = \tilde{\alpha}_r^2 \int_0^r \frac{g_s^2}{\tilde{\alpha}_s^2} \mathrm{d}s$). The probability flow ODE, corresponding to Eq. (5), which runs a "reverse" of the above process and has the same marginals p_t is

$$\mathrm{d}\boldsymbol{X}_t = \dot{k}(t) \left[a_t \boldsymbol{X}_t - \frac{g_t^2}{2} \nabla_{\boldsymbol{X}_t} \log p_t(\boldsymbol{X}_t) \right] \mathrm{d}t. \tag{7}$$

We can also define a reverse SDE with the same marginals as the original SDE, equivalent to combining probability flow with Langevin dynamics, which is most similar to the original formulation of sampling in diffusion models.

C LLM-FP as Bayesian Learning on an NDP-Defined Prior

In this situation we examine the meaning of the product-of-experts under the idealised cases of exact sampling and then learning.

We recall the data-conditioned NDP probabilities $p_1(\mathcal{Y} \mid \mathcal{X}, \mathcal{D})$, the LLM experts $q^j(\boldsymbol{y}^j) \coloneqq q(\boldsymbol{y}^j \mid \boldsymbol{x}^j, \mathcal{D}, \mathcal{C})$, and our product-of-experts predictive distribution:

$$\pi_1(\mathcal{Y} \mid \mathcal{X}, \mathcal{D}, \mathcal{C}) \;\propto\; p_1(\mathcal{Y} \mid \mathcal{X}, \mathcal{D}) \prod_{j=1}^{m} q^j(\boldsymbol{y}^j). \tag{8}$$

It can be helpful to see this rewritten in the (Gibbs) variational form:

$$\pi_1(\cdot \mid \mathcal{X}, \mathcal{D}, \mathcal{C}) = \arg\min_{\rho} \left\{ \mathbb{E}_{\mathcal{Y} \sim \rho} \left[\sum_{j=1}^{m} -\log q^j(\boldsymbol{y}^j) \right] + \mathrm{KL}\left(\rho \,\|\, p_1(\cdot \mid \mathcal{X}, \mathcal{D})\right) \right\}, \tag{9}$$

which shows that the LLM-FP selects the distribution over target outputs that best matches the LLM (in terms of cross-entropy) constrained by a KL penalty towards the NDP's predictions alone.

We can go further if we assume our NDP is fully trained to the Bayesian prior (the infinite data and capacity limit). Let $\nu(\mathcal{Y}_1^{\text{data}}, \mathcal{Y} \mid \mathcal{X}^{\text{data}}, \mathcal{X})$ be the NDP training-data output distribution (our "prior" over function space, since in the noiseless setting the $y_i = f(x_i)$ are function evaluations at the x_i over draws of f). $\mathcal{Y}_1^{\text{data}} = (\boldsymbol{y}_1^{\text{data},i})_{i=1}^n$ are the model's output variables at the context inputs (distinguished here from the context outputs

$\mathcal{Y}^{\text{data}}$ we actually observe), used in the NDP data measurement model $p_{\text{meas}}(\boldsymbol{y}^{\text{data},i} \mid \boldsymbol{y}_1^{\text{data},i})$. Excluding the $\mathcal{X}$ and $\mathcal{X}^{\text{data}}$ for notational brevity, the LLM-FP target is:

$$\pi_1\left(\mathcal{Y}_1^{\text{data}}, \mathcal{Y} \,\middle|\, \mathcal{D}, \mathcal{C}\right) \;\propto\; \nu\left(\mathcal{Y}_1^{\text{data}}, \mathcal{Y}\right) \prod_{i=1}^{n} p_{\text{meas}}\left(\boldsymbol{y}^{\text{data},i} \,\middle|\, \boldsymbol{y}_1^{\text{data},i}\right) \prod_{j=1}^{m} q^j\left(\boldsymbol{y}^j\right). \tag{10}$$

So we simply output from the prior over functions weighted by evidence from the p_{meas} measurement model and the q^j LLM experts. With the Gaussian measurement model we use and without experts this is simply the conditioning target of *e.g.* Song et al. [55].

In variational form with the Gaussian measurement model $p_{\text{meas}}(\boldsymbol{y}^{\text{data},i} \mid \boldsymbol{y}_1^{\text{data},i}) = \mathcal{N}(\boldsymbol{y}^{\text{data},i}; \boldsymbol{y}_1^{\text{data},i}, \sigma^2_{\text{meas}})$, we obtain

$$\pi_1(\cdot \mid \mathcal{D}, \mathcal{C}) = \arg\min_{\rho} \left\{ \mathbb{E}_{(\mathcal{Y}_1^{\text{data}}, \mathcal{Y}) \sim \rho} \left[\underbrace{\sum_{i=1}^{n} \frac{\left(\boldsymbol{y}^{\text{data},i} - \boldsymbol{y}_1^{\text{data},i}\right)^2}{2\sigma^2_{\text{meas}}}}_{\text{measurement nll}} + \underbrace{\sum_{j=1}^{m} -\log q^j(\boldsymbol{y}^j)}_{\text{text cross-entropy}} \right] + \underbrace{\mathrm{KL}(\rho \,\|\, \nu)}_{\text{functional prior}} \right\} \tag{11}$$

which exactly shows that an optimally trained LLM-FP selects a distribution over outputs based on optimising the likelihood under the data and experts subject to a penalty pulling us towards the functional prior the NDP was trained on.

We note the above is simply Bayesian learning on a prior defined by ν, with updating via the measurement model and via the 'virtual evidence' of the expert q [42].

D Additional Training Details for NDP

We parametrise our NDP flow network with a standard DiT [43], with the 'tokens' or 'patches' being concatenated inputs $(\boldsymbol{x}^i, \boldsymbol{y}_t^i)$ and outputs being the flow corresponding to each $\boldsymbol{y}_t^i$. As in the above, time is encoded using a sinusoidal embedding scheme and conditioned on through the adaptive Layer Norm-zero (adaLN-zero) scheme. As recommended we initialise the final layer weights to zero. Since our low-dimensional inputs do not interact well with layer normalisation, these inputs are initially passed through an embedding layer with GeLU [20] activation before the first transformer block.

Similarly to Dutordoir et al. [10], we train on a synthetic dataset of Gaussian Process samples of mixed kernel types and length-scales. Data is generated on-the-fly during training so that every minibatch is freshly sampled and the model never sees the same example twice. Each training example draws 256 (x, y) points with $x \in [-2, 2]$ (sampled stratified-uniform over equal-width bins) and y supported on $[-3.5, 3.5]$. With equal probability, the GP kernel is drawn from one of two branches:

RBF-only: a squared-exponential kernel with a short length-scale,

$$k_{\text{rbf}}(x, x') \;=\; \exp\!\left(-\tfrac{1}{2}\,\frac{(x - x')^2}{\ell_{\text{r}}^2}\right), \qquad \ell_{\text{r}} \sim \mathcal{U}[0.1,\ 1.0];$$

Locally-periodic: the product of an exp-sine-squared periodic kernel and a long-scale RBF envelope,

$$k_{\text{lp}}(x, x') \;=\; \exp\!\left(-\frac{2\sin^2\!\left(\pi(x - x')/p\right)}{\ell_{\text{p}}^2}\right) \cdot \exp\!\left(-\tfrac{1}{2}\,\frac{(x - x')^2}{\ell_{\text{R}}^2}\right),$$

with $p \sim \mathcal{U}[0.75,\ 1.25]$, $\ell_{\text{p}} \sim \mathcal{U}[0.5,\ 1.0]$, and $\ell_{\text{R}} \sim \mathcal{U}[1.0,\ 4.0]$.

Both branches use unit signal variance. This 50/50 mixture gives the model both smooth long-trend behaviour and (locally) seasonal structure at test time. We train the resulting unconditional NDP with an Optimal Transport flow matching objective [1, 33, 35] using AdamW [36] for 300,000 steps at batch size 2046 with a constant learning rate of 3×10^{-4}. We do not conduct large-scale scaling, extensive hyper-parameter search, or employ more complex data generation methods which could be used for improved results on real-world data of the NDP alone. We consider these out-of-scope for demonstrating the utility of our LLM-FP method combining NDPs with LLMPs (for example, 10 also consider other synthetic tasks like learning step functions to demonstrate that non-GP-like function spaces can be modelled, and 23 consider much more real-data-like structural causal models for data generation).

D.1 Validation

We perform two sanity checks on the trained NDP. Figure 6 verifies that unconditional samples are GP-like, and Figure 7 verifies that conditional samples produce sensible predictive distributions on both interpolation and extrapolation tasks, against a GP oracle baseline.

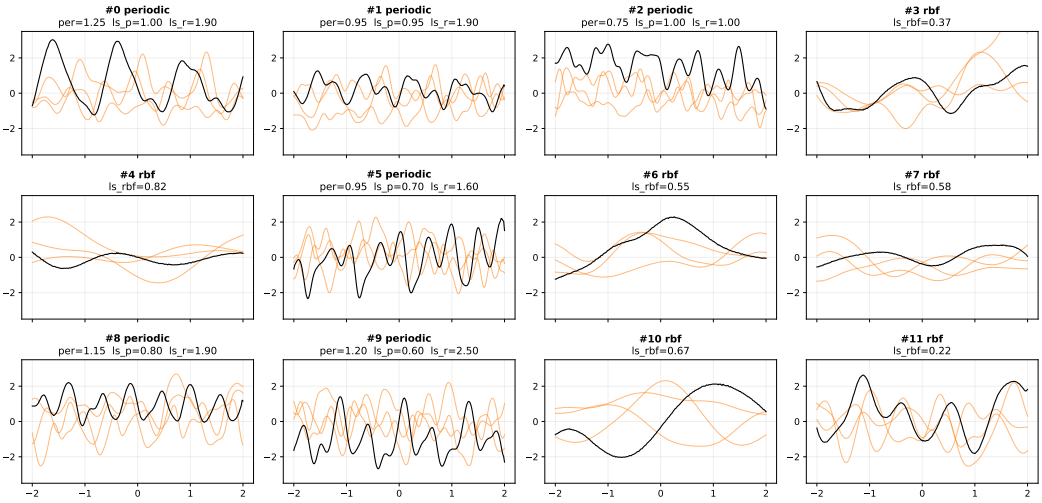

Figure 6: Unconditional NDP samples vs. fitted GP samples. Black: unconditional samples drawn from the trained NDP. Orange: samples from a GP whose kernel and hyperparameters were fitted by maximum likelihood to each NDP sample (the inferred kernel branch and hyperparameters are shown above each panel). The two are visually consistent, indicating that individual NDP samples look like plausible GP draws.

E Computational Costs

NDP model training was conducted on a Nvidia GH200 GPU over about 30 hours. We estimate an additional 100 hours of preliminary experiments for optimising the NDP configuration. No LLM training or fine-tuning was conducted; the fine-tuning in Appendix J applies to the NDP only and took about 2 hours on the same hardware. Inference for all experiments including preliminary ones took about 100 hours of GPU time, overwhelmingly dominated by local LLM inference.

F Prompt format

All LLM calls in our experiments are completions against a chat-templated prompt to Qwen-3.6-27B [49]. The three LLM-based methods evaluated (*LLM-Independent*, *LLM-Direct*, and the LLM term inside our *LLM-FP* guidance) share the same system prompt and the same way of rendering observed (x, y) data into the user message. They differ only in (a) what the user asks for at the end of the message and (b) whether the assistant first writes a `<think>...</think>` block. Each forecasting task fixes a natural-language description, the numeric range of y, the observed context points, and the rendering of x values (integer index, fixed-precision real, or absolute date).

F.1 System prompt

The system prompt used in all experiments (with the exception of Appendix H) is identical across problems and methods:

```
You are a time-series forecasting assistant.

Workflow:
  - You are given a description of the time series and some observed (x, value)
  ↪  pairs.
  - Reason about the function ONCE at the start.
  - You will then be asked to forecast specific x values, one at a time. Reuse your
  ↪  earlier reasoning for each query.

Keep your reasoning to roughly 500 words. You don't need to be exhaustive - sketch
↪  the pattern, sanity-check briefly, and stop. If you find yourself re-deriving
↪  the same conclusion, enumerating cases, or second-guessing a value you already
↪  wrote down, stop and finish.
```

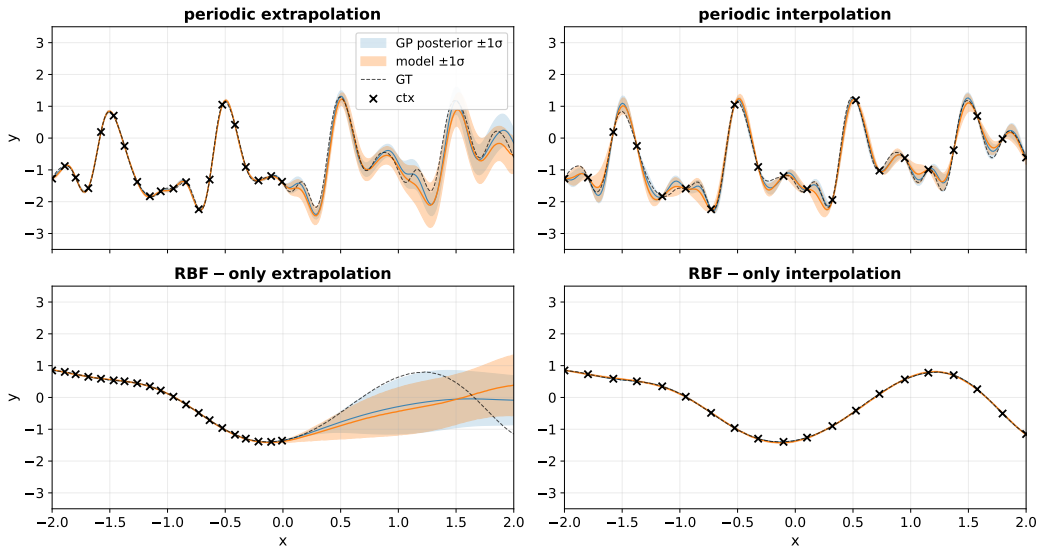

Figure 7: Conditional NDP samples vs. GP oracle. For four representative test functions (rows: locally-periodic and RBF), we condition the NDP on a set of context points (black) and show predictive marginals in orange. As an oracle reference, we also draw samples from a GP given the *true* kernel hyperparameters of the test function (blue). The NDP marginalises over its entire training prior, so it cannot match the oracle GP, but its predictive structure is consistent with the ground truth on both interpolation and extrapolation.

```
Internally reason step by step. Briefly cover these points:
  1. The real-world meaning of the values: what units, what is a typical magnitude,
  ↪  what is a plausible upper / lower bound.
  2. The structure of the series: trend (rising / falling), seasonality (daily /
  ↪  monthly / yearly cycle), regime changes (one-time events like a bankruptcy or
  ↪  holiday), or none of the above.
  3. The general shape of the function across the full x range. For seasonal data
  ↪  work out the cycle (for example, if month 1 is February then month 30 is (30 -
  ↪  1) mod 12 = 5 -> July, which is dry in California).
  4. A plan for how to map any queried x to its forecast value.

Once you have finished reasoning, your visible response is ONLY a single short
↪  acknowledgment sentence such as "I have analysed the data." or "Ready to
↪  forecast." Do not repeat, summarize, or restate your reasoning in the visible
↪  response. Then await the per-x queries. For each query, output ONLY the numeric
↪  forecast in the format `x, value`. No commentary, no units, no explanation.
```

F.2 User-message structure

Every user message has the same four parts:

1. A free-form natural-language description of the time series (what the data is, where it comes from, what is expected of it).
2. A *Constraints* block listing the y range with units and the x-domain rule.
3. The sorted observed (x, y) data lines, one per row, headed with a column label such as `day, Stock Price (GBP)`.
4. A query line. For per-target scoring the query asks for a single $x_\star$; for full-trajectory sampling it asks for T forecasts at a listed grid of target x values.

y values are rendered with banker's rounding (`decimal.ROUND_HALF_EVEN`) to a task-specific number of decimal places; the same precision determines the spacing of the discrete y grid used by per-target scoring. Context points are always sorted ascending in x before being inserted into the prompt.

F.3 Date vs. numeric x

How x is rendered depends on the task and falls into three cases:

- **Integer index** (e.g. Bankruptcy on day index $0, \ldots, 50$): the constraint line reads "Days are integers from 0 to 50" and each data line begins with a bare integer.
- **Fixed-precision real** (e.g. Periodic + linear trend on $x \in [-2, 2]$): the constraint line reads "Xs are real numbers in $[-2.00, 2.00]$" and x is formatted to a task-specific number of decimal places.
- **Absolute date** (Mauna Loa monthly / weekly): x is emitted as a `YYYY-MM-DD` date, the column header switches to "date", the constraint line reads "Dates are in YYYY-MM-DD (week-start, Sunday-anchored) format covering 2019-01-06 through 2023-11-26." (or similar), and the query reads "Forecast at 2021-07-04." rather than "Forecast at week 131."

F.4 LLM-Independent (per-target scoring)

For each forecast target $x_\star$ we build the user message described above with the query line (i.e. "Forecast at x="). We then apply Qwen's chat template to the conversation, append the literal string "$x_\star$, ", and score every candidate y on a regular grid (step matching the y-rendering precision) across the prompt y range. The score for each candidate is the sum of the LLM's token log-probabilities for the digits of y followed by a newline. Everything before the candidate y is shared across all candidates, so KV caching reduces each candidate to a short suffix forward pass. The per-grid scores are softmax-normalised over the y grid to give the marginal $\log p_{\mathrm{LLM}}(y \mid x_\star, \mathcal{D}_c)$. *LLM-Independent* samples each $x_\star$ independently from this marginal by inverse-CDF sampling; LLM-FP plugs the same per-target marginal into the guidance term of the NDP flow.

F.5 LLM-Direct (full-trajectory sampling)

LLM-Direct queries the model once for the entire forecast vector. The conversation is two user turns: a data-only primer identical to the one used for thinking mode (the assistant emits the `<think>...</think>` block plus a brief acknowledgement); then a single second user turn that explicitly lists the T target x values and demands T lines of output of the form `` `x, value` ``. Example A in Section F.7 shows both turns in full for the periodic + linear-trend preset. We sample $N = 800$ such trajectories with `do_sample=True` and constrain the decoder with `lmformatenforcer` against a regex of the form `<x_1>, -?\d+(?:\.\d+)?\n<x_2>, -?\d+(?:\.\d+)?\n...\n`, where the `<x_i>` are the literal target x values from the prompt, so that every completion is guaranteed to parse into a length-T vector with the prescribed x values in the correct order.

F.6 Thinking mode

For thinking-enabled runs we generate the `<think>...</think>` block exactly once per evaluation (per task and per random seed) by passing `enable_thinking=True` to the chat template, then re-use it across every per-target query (LLM-Independent, LLM-FP) or in front of the second user turn (LLM-Direct). Qwen-3.6 supports preserving a previously-emitted `<think>` block across subsequent turns via a `preserve_thinking=True` flag, which we use to share the thinking across queries. This sharing is what makes per-target scoring tractable: the model reasons about the whole function once (up to 4096 tokens of internal reasoning) and the per-target queries pay only the cost of scoring the y grid against a cached prefix.

F.7 Worked examples

Example A — Periodic + linear trend (real-valued x, LLM-Direct). Examples A and B illustrate two different methods on two different tasks: A shows LLM-Direct with full-trajectory sampling, B shows LLM-Independent / LLM-FP with the multi-turn thinking-mode path. The data-only first user turn is structurally the same in both cases. The preset for Example A uses 20 context points evenly spaced on $x \in [-2, 0]$ at full `float64` resolution (true spacing $2/19 \approx 0.10526$, which renders as alternating 0.10 / 0.11 steps at a 2-decimal-place rendering). The LLM is asked to forecast on the full domain $x \in [-2, 2]$ at $T = 51$ target locations, obtained by subsampling 51 of the NDP's 256 dense grid points via `np.linspace(0, 255, 51).round()`; the targets are therefore approximately but not exactly uniformly spaced (see Appendix G for the alignment constraint motivating this). In LLM-Direct mode the first user turn is data-only — it primes the model to think about the function but does not yet ask for any forecast value:

```
I have a time series forecasting task for you. Here's some context about the task:

A periodic function with a linear trend
```

```
Numerical constraints:
- Values are real numbers in [-3.5, 3.5].
- Xs are real numbers in [-2.00, 2.00].

Observed data (x, value):
-2.00, -1.69
-1.89, -0.54
-1.79, -0.04
-1.68, 0.52
-1.58, 0.09
-1.47, 0.20
-1.37, -0.95
-1.26, -1.24
-1.16, -0.70
-1.05, -0.76
-0.95, 0.50
-0.84, 0.57
-0.74, 1.01
-0.63, 0.76
-0.53, 0.24
-0.42, 0.22
-0.32, -0.37
-0.21, -0.10
-0.11, 0.18
0.00, 1.16

Think carefully about the function: its general shape across the full range, any
↪  seasonality, trends, regime changes, or events. After you have thought you do
↪  not need to respond, just acknowledge you are ready.

Each forecast you give will be in the format `x, value`.
```

The assistant then emits its single <think>...</think> block followed by a one-sentence acknowledgement. A second user turn then requests all $T = 51$ forecast values in one go:

```
Now produce the full forecast.

Forecast at the following xs (in this exact order):
-2.00, -1.92, -1.84, -1.76, -1.69, -1.59, -1.51, -1.44, -1.36, -1.28, -1.20, -1.12,
↪  -1.04, -0.96, -0.89, -0.81, -0.71, -0.64, -0.56, -0.48, -0.40, -0.32, -0.24,
↪  -0.16, -0.09, -0.01, 0.09, 0.16, 0.24, 0.32, 0.40, 0.48, 0.56, 0.64, 0.71,
↪  0.79, 0.89, 0.96, 1.04, 1.12, 1.20, 1.28, 1.36, 1.44, 1.51, 1.59, 1.69, 1.76,
↪  1.84, 1.92, 2.00

Reply with EXACTLY 51 lines and nothing else. Each line must be `x, value` --- the x
↪  from the list above, then a comma and a space, then the predicted value as a
↪  real number. Output the 51 lines in the same order as the list. Do not include
↪  headers, commentary, units, blank lines, or any text before or after the 51
↪  lines.
```

Example B — Mauna Loa weekly CO_2, 20 random context weeks (date x, LLM-Independent / LLM-FP, thinking mode). Same machinery, with x rendered as Sunday-anchored weekly dates and a custom date-range constraint line. To prevent memorisation, these dates are chosen at random and the corresponding CO_2 values are interpolated from the monthly data (*cf.* the subsection on memorisation in 63). In thinking mode the per-target query is a *multi-turn* conversation: a data-only primer (the same form as Example A, just with the Mauna Loa task description and observations), one assistant turn that contains the <think>...</think> block plus a brief "Ready." acknowledgement, and then *one short user turn per target* $x_\star$.

```
I have a time series forecasting task for you. Here's some context about the task:

Weekly mean atmospheric CO2 concentration in ppm at Mauna Loa Observatory across
↪  roughly 5 years (256 weekly observations from 2019-01-06 through 2023-11-26).
↪  The provided observations are 20 sparsely-and-randomly sampled weeks from the
↪  first half (through 2021-06-13); forecast the remainder.
```

```
Numerical constraints:
- Values represent CO2 (ppm), real numbers in [405.0, 430.0].
- Dates are in YYYY-MM-DD (week-start, Sunday-anchored) format covering 2019-01-06
↪  through 2023-11-26.

Observed data (date, CO2 (ppm)):
2019-01-13, 411.0
2019-02-03, 411.6
2019-03-03, 412.1
2019-05-26, 414.6
2019-08-04, 410.8
2019-09-01, 409.4
2020-02-02, 414.1
2020-03-01, 414.6
2020-04-26, 416.8
2020-05-10, 417.2
2020-05-17, 417.2
2020-06-07, 416.7
2020-06-28, 415.6
2020-07-12, 414.7
2020-09-27, 411.5
2020-10-11, 411.5
2020-11-01, 412.5
2021-02-07, 416.6
2021-04-18, 419.0
2021-06-13, 418.9

Think carefully about the function: its general shape across the full range, any
↪  seasonality, trends, regime changes, or events. After you have thought you do
↪  not need to respond, just acknowledge you are ready.

Each forecast you give will be in the format `date, value`.
```

The assistant then emits a `<think>...</think>` block (which we share across all 64 per-target queries) followed by a one-sentence acknowledgement (e.g. "Ready to forecast."). For *each* forecast date $x_\star$ we then append a single user turn of the form

```
Forecast at date=2021-07-04.
```

re-render the whole conversation through the chat template with `preserve_thinking=True`, `enable_thinking=False`, and prime the assistant turn with "`2021-07-04, `" so the next token is the first digit of y. LLM-Independent / LLM-FP construct the per-target marginal $\log p_{\mathrm{LLM}}(y \mid x_\star, \mathcal{D}_c)$ by scoring y on a 0.1-ppm grid spanning $[405, 430]$ against this prefix. KV caching means the system prompt, primer, and shared assistant-with-`<think>` turn are computed only once, and each per-target query reuses that cache — only the short "`Forecast at date=...`" suffix is recomputed per $x_\star$.

The per-target queries are issued at 64 dates subsampled from the 256-week NDP grid at evenly spaced indices, spanning the full x range and not just the forecast region: 32 sit inside the context window and 32 sit in the forecast region. The LLM-FP guidance therefore conditions the NDP at 64 anchor points. The corresponding LLM-Direct prompt is the same primer followed by the *single* batched user turn shown in Section F.5, requesting all 64 dates at once.

G LLM–NDP target alignment

The LLM-FP guidance term in Eq. (3) couples the LLM marginal $\log \widetilde{q}_{r_t}(\widehat{\boldsymbol{y}}_{1|t} \mid c)$ to the NDP's expected denoised output $\widehat{\boldsymbol{y}}_{1|t}$ at the same target locations. Concretely, if the LLM is queried at x value $x_\star$ but the NDP is sampling at a different set of x values, there is no $\widehat{\boldsymbol{y}}_{1|t}$ to evaluate the LLM marginal against, and the guidance term is undefined.

In practice, our NDP transformer takes x values as part of its input tokens and can in principle predict at any chosen set of locations, so this is not a fundamental constraint. Each LLM target is anchored to a single NDP grid position, for two reasons. First, where the LLM targets are chosen as a subset of the NDP grid, this lets the LLM marginal at $x_\star$ enter the guidance for the NDP's prediction at exactly that $x_\star$, without interpolation between LLM anchor points. Second, querying the LLM at every NDP grid point would be expensive (the

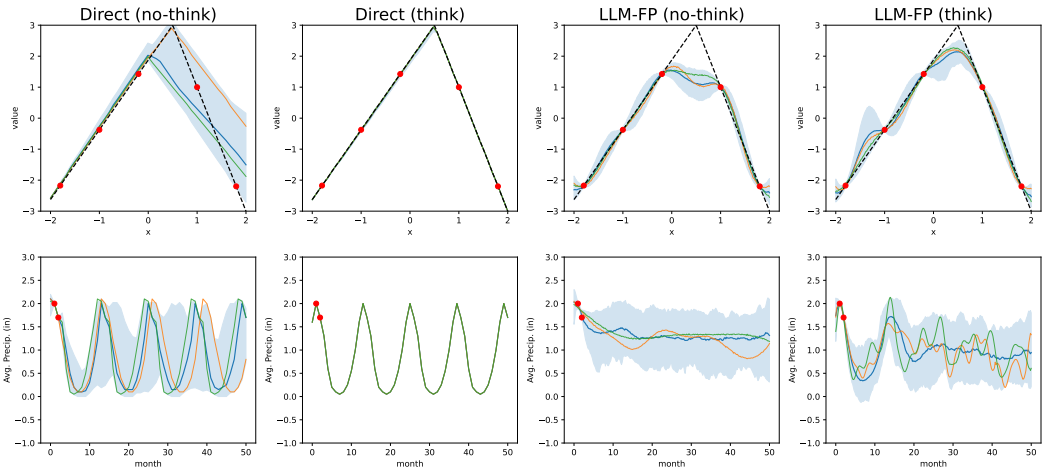

Figure 8: Effect of enabling LLM reasoning ("thinking") on the Change point (top) and San Diego (bottom) tasks. Reasoning generally improves consistency and predictive accuracy across the LLM-based methods, but can also reduce predictive uncertainty: on San Diego, LLM-Direct with reasoning collapses to near-identical samples.

per-target scoring cost is dominated by the number of LLM queries); aligning a sparser subset of LLM targets to a denser NDP grid lets us trade off LLM cost against NDP resolution. The 51 LLM targets in Example A and the 64 LLM dates in Example B (see Appendix F.7) are obtained by subsampling the NDP's 256-point grid at uniform indices. The remaining grid positions are left uninformative.

H Thinking vs no thinking

All results in this paper use thinking-enabled prompts, where the LLM emits a `<think>...</think>` reasoning block before forecasting (Section F.6). For completeness we examine the effect of disabling thinking, replacing the per-target query path with a data-only single-turn variant and dropping the `<think>` block.

On Change point, both LLM-Direct and LLM-FP fail to capture the change point without thinking: LLM-Direct picks linear continuations that miss the true function, and LLM-FP smooths out the peak at the intersection of the two segments. With thinking, both recover the change-point structure.

San Diego shows a different trade-off. Without thinking, LLM-Direct produces samples that each individually repeat a fixed pattern but vary across samples, giving better predictive diversity than the thinking-enabled version, where all samples collapse onto a single periodic template. LLM-FP without thinking retains almost no structure, with samples reverting to the NDP prior; with thinking it concentrates on more plausible trajectories.

Thinking helps when the model can lock onto structure (Change point) and can hurt when it leaves the autoregressive rollout with insufficient residual variability (LLM-Direct on San Diego). LLM-FP is less exposed to the latter: per-target marginals retain more variability than autoregressive rollouts even after a confident reasoning pass, for the reasons discussed in Section 6.

I Additional Experiments and Details

I.1 Metrics Definitions

We report three metrics for every method on every task: mean absolute error (MAE), continuous ranked probability score (CRPS) in its empirical-ensemble form, and negative log predictive density (NLPD) under a Gaussian fit to the ensemble. Each method produces an ensemble of N trajectories $\{y^{(i)}\}_{i=1}^{N}$, where each $y^{(i)} \in \mathbb{R}^T$ is a sample over the evaluation grid $x_1, \ldots, x_T$ used for that task (whose size and extent depend on the task), and the ground truth is a deterministic function $y^\star(x)$. We use $N = 800$ for all methods. All three metrics are reported in the original y units of each task (no normalisation) and are oriented so that lower values indicate better forecasts. The average is taken over the *full evaluation grid*, including positions inside the context region, because we are interested in the performance of methods for interpolation as well as extrapolation.

MAE. The MAE of the per-position ensemble mean against the truth,

$$\mathrm{MAE} \;=\; \frac{1}{T}\sum_{t=1}^{T}\bigl|\bar{y}_t - y^\star(x_t)\bigr|, \qquad \bar{y}_t \;=\; \frac{1}{N}\sum_{i=1}^{N} y_t^{(i)}.$$

This summarises the point-forecast accuracy of the ensemble's centre and reduces to ordinary MAE for a deterministic predictor.

CRPS. The empirical-ensemble form of CRPS [13],

$$\mathrm{CRPS} \;=\; \frac{1}{T}\sum_{t=1}^{T}\left[\frac{1}{N}\sum_{i=1}^{N}\bigl|y_t^{(i)} - y^\star(x_t)\bigr| \;-\; \frac{1}{2N^2}\sum_{i=1}^{N}\sum_{j=1}^{N}\bigl|y_t^{(i)} - y_t^{(j)}\bigr|\right].$$

This is a strictly proper score that rewards both calibration and sharpness, and collapses to MAE when the ensemble is a single repeated point.

NLPD. For each evaluation position t we fit a Gaussian $\mathcal{N}(\mu_t, \sigma_t^2)$ to the ensemble and report the average negative log density of the ground-truth value under that fit:

$$\mathrm{NLPD} \;=\; \frac{1}{T}\sum_{t=1}^{T}\left[\tfrac{1}{2}\log\bigl(2\pi\sigma_t^2\bigr) \;+\; \frac{\bigl(y^\star(x_t) - \mu_t\bigr)^2}{2\,\sigma_t^2}\right],$$

where

$$\mu_t \;=\; \frac{1}{N}\sum_{i=1}^{N} y_t^{(i)}, \qquad \sigma_t \;=\; \sqrt{\frac{1}{N-1}\sum_{i=1}^{N}\bigl(y_t^{(i)} - \mu_t\bigr)^2}.$$

We use the Bessel-corrected (`ddof=1`) sample standard deviation. The logarithm is natural, so NLPD is reported in nats. To prevent divergence on positions where the ensemble has collapsed to a near-deterministic point, we clip σ_t from below at $\sigma_{\min} = 10^{-3}$ in the original y units before evaluating the formula. This is a numerical guard, not a model assumption: it caps the spurious $\log\sigma$ savings of a degenerate ensemble while still penalising confidently-wrong predictions through the $(y^\star - \mu)^2/2\sigma_{\min}^2$ term.

Uncertainty estimation. Each cell in the results table is reported as $m_{\pm h}$, where m is the metric computed as defined above and $h = (q_{97.5} - q_{2.5})/2$ is half the width of the 95% bootstrap confidence interval. The bootstrap resamples the T evaluation positions of the cell with replacement, repeats this $B = 1000$ times, and takes the 2.5% and 97.5% quantiles of the resulting metric distribution. This captures uncertainty arising from *which* test points the metric was averaged over rather than Monte Carlo noise from the predictive ensemble, which a separate sample-bootstrap shows to be typically 3–15$\times$ tighter at $N = 800$ and is therefore not reported.

I.2 Additional qualitative comparisons

Figures 9 and 10 show two additional tasks, taken from [50]: a synthetic bankruptcy scenario where the prompt specifies that the value goes to zero at day 30, and Montreal daily temperature extrapolation with a textual description of the location and variable.

Bankruptcy is a test of textual conditioning: the day-30 constraint is unrecoverable from the numerical context alone. All text-conditioned methods (LLM-Direct, LLM-Independent, LLM-FP) respect the constraint, while the unconditional NDP extrapolates from its training prior and ignores it.

Montreal is a softer setting where all text-conditioned methods produce broadly reasonable predictions. LLM-Direct samples tend to follow near-linear trajectories not matching those suggested by the LLM marginals; LLM-FP combines the LLM marginals with the NDP's local structure to produce smoother seasonal trajectories.

We do not evaluate on the Context-is-Key benchmark [63], from which our LLM-Direct baseline is adapted, as it targets a different question from ours. Its tasks are hand-crafted so that correctly using the textual context is the primary factor determining performance. Where needed, the underlying series are edited to match the context, for example by injecting short multiplicative shocks into the forecast horizon (their Sec. B.2), and other tasks feature abrupt regime changes such as piecewise-constant inputs (Sec. B.6). The resulting targets are often highly non-smooth. Moreover, over a third of the tasks (29 of 71) provide at most 6 historical data points. Success therefore depends primarily on text comprehension and on reproducing deliberate discontinuities, rather than on generating long, coherent trajectories. Our method instead aims to produce smooth, internally consistent joint samples by combining LLM conditioning with an NDP prior. A prior of this kind would place little mass on such discontinuities, and we would not expect it to add much on tasks that hinge on reading the text. In principle, the prior could be extended to cover such behaviour by fine-tuning on similar data, or by training on rich synthetic distributions in the style of prior-fitted networks. The former is not possible on CiK itself, which is an evaluation-only benchmark with no training split. We leave this to future work.

:ompany's daily stock price in GBP over 50 trading days. On day 30 the company enters compulsory liquidation and is p
from the London Stock Exchange."

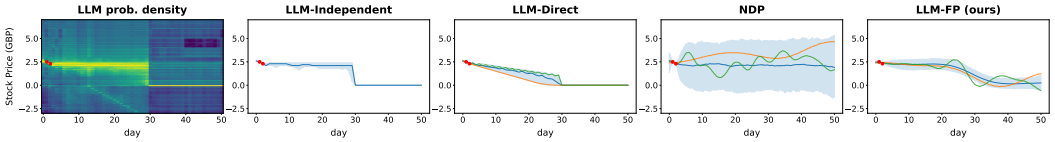

Figure 9: Synthetic bankruptcy scenario. The text-conditioned methods follow the day-30 constraint specified in the prompt; the NDP extrapolates from its training prior.

Prompt: "Daily average temperature in Montreal in degrees Celsius over 50 days in May and June, starting May 1st."

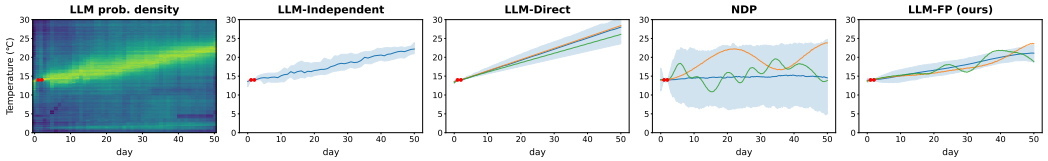

Figure 10: Daily temperature extrapolation in Montreal. All text-conditioned methods produce reasonable predictions; LLM-Direct samples tend to follow near-linear trajectories rather than the seasonal structure suggested by the LLM marginals.

J Fine-tuning on Real Mauna Loa Data

To give an idea of how our method could perform with a stronger NDP (i.e. one meta-learned across many real and synthetic datasets, as in Hollmann et al. [24]), we add an experiment in which the NDP is fine-tuned on real data from Mauna Loa. We fine-tune a single model jointly on four series, two harder (CO_2 [45] and CH_4) and two easier (N_2O and SF_6), for 3000 steps in total. The latter three series are taken from the NOAA surface-flask record at Mauna Loa [29–31]. For CO_2 we use the raw weekly series rather than the weekly series interpolated from monthly data used in the main paper; the raw data is considerably noisier and so falls outside the support of the simple synthetic prior of the un-fine-tuned NDP. Each training example is a single time window from one of these series, randomly sliced from the historical data up to a cutoff just before the prediction window. At evaluation we draw 20 sparse random observations from the first half of each window as context and score each method across the entire window. Table 2 reports the results and Fig. 11 shows the corresponding predictions. With the fine-tuned prior the NDP is already strong on these series and as a result the benefit from text conditioning narrows. LLM-FP is the best method on CO_2 and SF_6 in MAE and CRPS, while the NDP alone is best throughout on CH_4 and N_2O. This is expected, as the NDP's prior already captures their seasonality and trend, leaving little for the textual description to add.

Table 2: Predictive performance after fine-tuning on real Mauna Loa data, with a sparse context of 20 scattered observations from the first half of each series; lower is better, best per row in bold. The LLM-Direct ensemble collapsed to a few trajectories, which is why its NLPD is very high for CH_4 and SF_6.

Series	Metric	LLM-Independent	LLM-Direct	NDP	LLM-FP (ours)
CO_2 2019	MAE	0.679	1.366	0.722	**0.504**
	CRPS	0.526	1.112	0.555	**0.380**
	NLPD	**1.447**	3.584	2.151	1.570
CH_4 2009	MAE	5.645	5.038	**2.978**	3.816
	CRPS	5.000	4.480	**2.214**	2.922
	NLPD	5.217	2.2×10^4	**2.669**	3.255
N_2O 2020	MAE	0.172	0.079	**0.052**	0.056
	CRPS	0.114	0.055	**0.037**	0.040
	NLPD	−0.372	0.252	**−1.577**	−1.455
SF_6 2020	MAE	0.036	0.025	0.025	**0.023**
	CRPS	0.023	0.023	0.018	**0.017**
	NLPD	−1.245	93.435	**−2.242**	−2.197

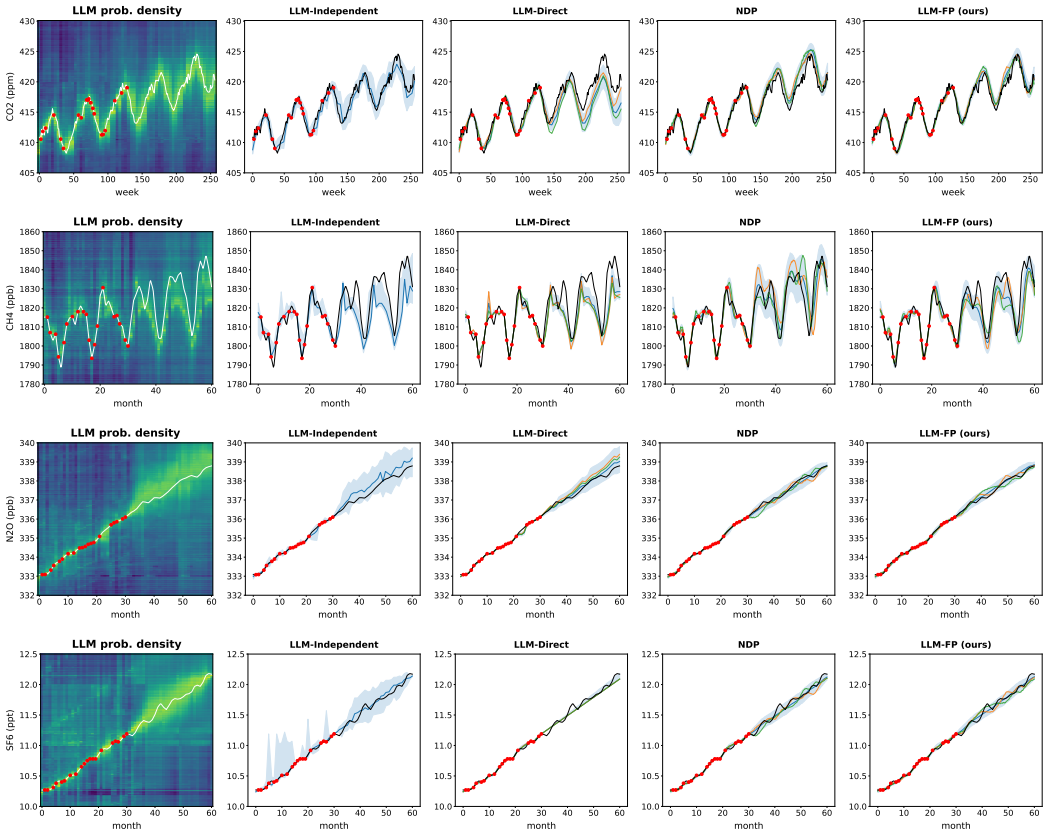

Figure 11: Predictions after fine-tuning on real Mauna Loa data, for the four gas series of Table 2 (rows: CO_2 weekly, CH_4, N_2O and SF_6 monthly). The leftmost column shows the LLM marginal density used by LLM-Independent and LLM-FP; context observations are in red and the ground truth in black. On the seasonal CO_2 and CH_4 series, LLM-Direct tracks the context but drifts in amplitude and is badly overconfident, giving an essentially divergent NLPD on CH_4. LLM-FP is the most accurate method on CO_2, while on CH_4 the fine-tuned NDP is already the best of the four. On the near-monotonic N_2O and SF_6 series all methods do well and the NDP alone is close to the truth throughout.

K Weaker LLM Ablation

Table 3 compares the results of Table 1 using Qwen-3.6-27B with the same experiments using the smaller Qwen-3.5-9B LLM.

We reproduce the setup of Table 1 with the smaller LLM in Table 3. Every LLM-based method degrades. The smaller model responds much less reliably to the textual information (as can be seen in Figure 12). LLM-FP nonetheless remains the best of the LLM-based methods on all task–metric combinations, but with the weaker LLM it no longer reliably improves on the unconditioned NDP, doing so in only six of the nine.

L Effect of the Number of Context Points

For this experiment, we randomly sub-sample different numbers of context points for our Mauna Loa CO_2 task and examine how varying the number of points affects forecasts (with the evaluation over the full grid); results are given in Table 4. As expected, the LLM textual conditioning is strongest when there are fewer points, whereas the NDP performs better with more numerical context. From Table 4, from $n = 10$ onwards LLM-FP is the best method on every metric, outperforming both of its components.

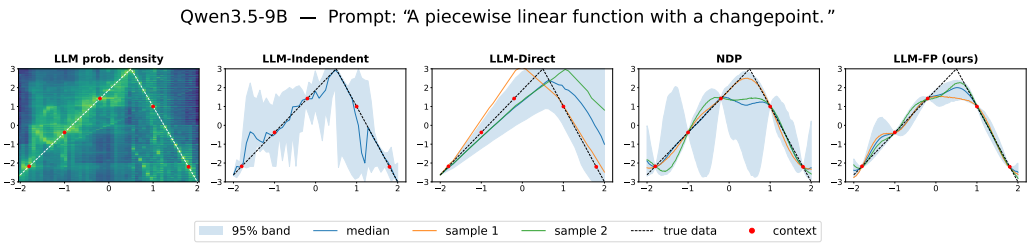

Figure 12: Change-point regression with the smaller LLM, to be compared with Fig. 2. LLM-Independent and LLM-Direct capture the change point to an extent, but with wide uncertainty bands. LLM-FP degrades more gracefully, staying closest to the true function (Table 3).

Table 3: Effect of a weaker LLM expert on predictive performance; lower is better. Each LLM-based cell shows the result with Qwen-3.6-27B $\rightarrow$ Qwen-3.5-9B. The NDP does not use an LLM and so has a single value. Best per row (using the Qwen-3.5-9B results) in bold.

Task	Metric	LLM-Independent	LLM-Direct	NDP	LLM-FP (ours)
Change point	MAE	$0.287 \rightarrow 0.469$	$0.035 \rightarrow 0.717$	0.526	$0.129 \rightarrow \mathbf{0.244}$
	CRPS	$0.118 \rightarrow 0.269$	$0.013 \rightarrow 0.434$	0.305	$0.100 \rightarrow \mathbf{0.175}$
	NLPD	$0.055 \rightarrow 0.719$	$-1.770 \rightarrow 0.839$	0.619	$-0.230 \rightarrow \mathbf{0.058}$
Periodic + linear	MAE	$0.267 \rightarrow 0.519$	$0.573 \rightarrow 0.619$	**0.272**	$0.152 \rightarrow 0.332$
	CRPS	$0.219 \rightarrow 0.358$	$0.420 \rightarrow 0.436$	**0.183**	$0.103 \rightarrow 0.235$
	NLPD	$0.237 \rightarrow 0.663$	$0.887 \rightarrow 1.119$	$\mathbf{-0.393}$	$-0.731 \rightarrow -0.316$
Mauna Loa CO_2 (weekly)	MAE	$0.596 \rightarrow 0.976$	$1.051 \rightarrow 3.406$	0.867	$0.514 \rightarrow \mathbf{0.649}$
	CRPS	$0.427 \rightarrow 0.764$	$0.805 \rightarrow 2.763$	0.573	$0.362 \rightarrow \mathbf{0.449}$
	NLPD	$0.914 \rightarrow 1.558$	$1.949 \rightarrow 4.773$	0.923	$0.690 \rightarrow \mathbf{0.833}$

M Quantifying Function-Class Selection

Table 5 quantifies the function-class selection discussed in Section 6 on the San Diego task of Fig. 3, where the text describes seasonal structure that the numerical context alone leaves undetermined. Conditioning on the LLM probabilities raises the share of samples assigned to the periodic branch and the share whose dominant period is close to a 12-month season.

Both quantities come from a single GP model-selection fit over 800 samples per method per task, each evaluated on the preset's 256-point grid spanning months 0–50; evaluated for two context sizes (2 and 3 observed points). Every sample is scored under the two kernel branches from the NDP's training distribution: periodic $\times$ RBF over a grid $\mathcal{G}$ of 128 periods $\times$ 32 periodic $\times$ 32 RBF lengthscales, and RBF-only over 128 lengthscales. We then compute exact GP log marginal likelihood with observation noise $\sigma = 0.05$. *Periodic* is the fraction of samples whose branch posterior favours the periodic branch ($w_{\text{per}} > 0.5$). *Period $\approx$ 12 mo* is the fraction whose maximum-likelihood period over $\mathcal{G}$ lies within $\pm 10\%$ of 12 months.

With two context points the periodic share rises substantially ($0.454 \rightarrow 0.704$) while the period share improves only modestly ($0.166 \rightarrow 0.275$). In this case the text is enough to select periodic structure, but not to fix its frequency. Figure 3 suggests why: with only two observations the LLM marginals retain enough uncertainty over the first 12 months to leave the period of the LLM-FP's samples loosely constrained. With a third observation they tighten and the period becomes more accurate, with 0.736 of samples falling within $\pm 10\%$ of a 12-month season. Consistent with Appendix K, where a smaller LLM degrades performance, we expect a stronger LLM to produce sharper marginals from two observations and so pin the period down further.

Table 4: Predictive performance on the Mauna Loa CO_2 (weekly) task as the number of context points n varies. Each cell is the mean over three independent random context draws, with the min–max range across draws as a subscript. At $n = 20$ the context is the entire pool, so a single run is reported; this is a separate run from Table 1, so the LLM-based values differ slightly. Lower is better, best per row in bold.

Context points	Metric	LLM-Independent	LLM-Direct	NDP	LLM-FP (ours)
$n = 3$	MAE	$\mathbf{1.14}_{[\mathbf{0.98}, \mathbf{1.29}]}$	$1.21_{[0.79, 1.73]}$	$2.21_{[2.19, 2.25]}$	$1.30_{[0.93, 1.73]}$
	CRPS	$0.85_{[0.70, 0.98]}$	$\mathbf{0.84}_{[\mathbf{0.67}, \mathbf{1.05}]}$	$1.64_{[1.56, 1.71]}$	$0.99_{[0.66, 1.42]}$
	NLPD	$2.03_{[1.81, 2.18]}$	$\mathbf{1.78}_{[\mathbf{1.66}, \mathbf{1.87}]}$	$2.39_{[2.31, 2.50]}$	$2.65_{[1.56, 4.40]}$
$n = 5$	MAE	$\mathbf{1.09}_{[\mathbf{0.84}, \mathbf{1.31}]}$	$1.43_{[1.10, 1.93]}$	$2.30_{[2.06, 2.52]}$	$1.30_{[0.86, 1.73]}$
	CRPS	$\mathbf{0.80}_{[\mathbf{0.62}, \mathbf{0.94}]}$	$1.04_{[0.83, 1.38]}$	$1.65_{[1.56, 1.77]}$	$0.99_{[0.60, 1.43]}$
	NLPD	$\mathbf{1.76}_{[\mathbf{1.50}, \mathbf{1.91}]}$	$2.13_{[1.89, 2.28]}$	$2.35_{[2.32, 2.37]}$	$2.61_{[1.37, 4.50]}$
$n = 10$	MAE	$0.92_{[0.75, 1.12]}$	$1.38_{[1.02, 1.73]}$	$1.90_{[1.71, 2.11]}$	$\mathbf{0.77}_{[\mathbf{0.67}, \mathbf{0.93}]}$
	CRPS	$0.67_{[0.55, 0.82]}$	$0.98_{[0.68, 1.28]}$	$1.24_{[1.11, 1.38]}$	$\mathbf{0.55}_{[\mathbf{0.47}, \mathbf{0.68}]}$
	NLPD	$1.55_{[1.28, 1.86]}$	$1.87_{[1.50, 2.11]}$	$1.74_{[1.67, 1.86]}$	$\mathbf{1.17}_{[\mathbf{1.02}, \mathbf{1.44}]}$
$n = 15$	MAE	$0.72_{[0.70, 0.75]}$	$1.51_{[1.16, 1.86]}$	$1.03_{[0.98, 1.08]}$	$\mathbf{0.63}_{[\mathbf{0.60}, \mathbf{0.67}]}$
	CRPS	$0.52_{[0.50, 0.54]}$	$1.12_{[0.83, 1.44]}$	$0.68_{[0.66, 0.72]}$	$\mathbf{0.45}_{[\mathbf{0.41}, \mathbf{0.50}]}$
	NLPD	$1.22_{[1.16, 1.27]}$	$2.05_{[1.51, 2.77]}$	$1.15_{[1.11, 1.19]}$	$\mathbf{0.87}_{[\mathbf{0.78}, \mathbf{1.04}]}$
$n = 20$	MAE	0.65	1.31	0.87	**0.50**
	CRPS	0.48	0.96	0.57	**0.37**
	NLPD	1.20	1.73	0.92	**0.75**

Table 5: Function selection on the San Diego task. Both rows come from the same GP marginal-likelihood fit over a 128×32×32 grid of (period, ℓ_{per}, ℓ_{rbf}): *Periodic* is the fraction of samples whose periodic-branch posterior exceeds 0.5, and *Period ≈ 12 mo* the fraction whose maximum-likelihood period lies within ±10% of a 12-month season. 800 samples per method per task.

Task	Quantity	NDP	LLM-FP (ours)
San Diego (2 pts)	Periodic	0.454	**0.704**
	Period ≈ 12 mo	0.166	**0.275**
San Diego (3 pts)	Periodic	0.591	**0.831**
	Period ≈ 12 mo	0.284	**0.736**

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The abstract and introduction clearly state claims and have a contributions sections which are justified by the paper.

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: We include a limitations section in the Discussion (Section 6)

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [Yes]

   Justification: The mathematical contribution in Section 3 is a constructive derivation rather than a formal theorem; the approximation used is sourced and cited from Pokle et al. [46], Song et al. [55] with other intermediate steps shown.

   Guidelines:

   - The answer [N/A] means that the paper does not include theoretical results.

- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: The method is described in Sections 3 and 4 and the experimental setup in Section 5. NDP architecture and training are detailed in Appendix D, LLM prompts, decoding and target alignment in Appendices F and G, and metrics in Appendix I.1. Additional experiments are described in Appendices J to M. Code is available at https://github.com/sam-willis/LLM-flow-processes.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
     (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [Yes]

   Justification: The codebase is released at https://github.com/sam-willis/LLM-flow-processes; the atmospheric datasets are publicly available from NOAA [45, 29–31].

   Guidelines:

   - The answer [N/A] means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (https://neurips.cc/public/guides/CodeSubmissionPolicy) for more details.

- While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (https://neurips.cc/public/guides/CodeSubmissionPolicy) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

   Answer: [Yes]

   Justification: NDP architecture, synthetic training data, optimiser, learning rate, batch size and number of training steps are given in Appendix D, with fine-tuning details in Appendix J. LLM prompts, decoding constraints and per-target scoring are given in Appendices F and G, and evaluation metrics in Appendix I.1. No LLM is trained or fine-tuned.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

   Justification: We report bootstrap confidence intervals over evaluation positions in Table 1 with the procedure described in Appendix I.1. Table 4 reports the mean and min–max range over three independent random draws of the context set. The results in Tables 2, 3 and 5 are point estimates only.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
   - The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
   - The assumptions made should be given (e.g., Normally distributed errors).
   - It should be clear whether the error bar is the standard deviation or the standard error of the mean.
   - It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
   - For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
   - If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

Answer: [Yes]

Justification: We provide an estimate of overall compute usage including in preliminary experiments in Appendix E.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
- The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
- The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics https://neurips.cc/public/EthicsGuidelines?

Answer: [Yes]

Justification: We have reviewed the code of conduct and our paper does not intersect with the harms outlined there.

Guidelines:

- The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
- If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
- The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

Answer: [N/A]

Justification: The work is foundational research on probabilistic regression and conditional diffusion sampling. We do not foresee direct negative societal impacts beyond those generic to improvements in LLM-based forecasting.

Guidelines:

- The answer [N/A] means that there is no societal impact of the work performed.
- If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

Answer: [N/A]

Justification: The paper does not release pre-trained generative models, scraped datasets, or other artefacts with elevated misuse risk. The NDP is trained on synthetic Gaussian Process samples; the LLMs (Qwen3.6-27B and Qwen3.5-9B) used are frozen, externally-released models.

Guidelines:

- The answer [N/A] means that the paper poses no such risks.
- Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
- Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
- We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

Answer: [Yes]

Justification: We use Qwen3.6-27B [49] and Qwen3.5-9B [48] (both Apache 2.0 License), and the Mauna Loa CO_2 [45], CH_4 [29], N_2O [30] and SF_6 [31] records from NOAA GML (public domain, CC0 License). All references are cited in the relevant sections.

Guidelines:

- The answer [N/A] means that the paper does not use existing assets.
- The authors should cite the original paper that produced the code package or dataset.
- The authors should state which version of the asset is used and, if possible, include a URL.
- The name of the license (e.g., CC-BY 4.0) should be included for each asset.
- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

Answer: [Yes]

Justification: We release the code for LLM-FP and all baselines at `https://github.com/sam-willis/LLM-flow-processes` under the Apache-2.0 licence.

Guidelines:

- The answer [N/A] means that the paper does not release new assets.
- Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
- The paper should discuss whether and how consent was obtained from people whose asset is used.
- At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

Answer: [N/A]

Justification: No crowdsourcing or human-subjects research.

Guidelines:

- The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
- Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
- According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

Answer: [N/A]

Justification: No crowdsourcing or human-subjects research.

Guidelines:

- The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

Answer: [Yes]

Justification: LLMs are a core part of the method: Qwen3.6-27B is used as the expert in our LLM-FP method and as part of the LLM-Direct and LLM-Independent baselines, with the smaller Qwen3.5-9B used for the ablation in Appendix K. Prompt specifications, decoding constraints, and target alignment are detailed in Appendices F and G.

Guidelines:

- The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
- Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.