ATEMPO: AMORTIZED TRANSFER ENTROPY MODEL PER-LAG OVER-TIME

Anonymous authors
Paper under double-blind review

ABSTRACT

When one time series informs another, the delay at which that information arrives determines if models can utilize this relationship effectively. A source may be strongly predictive at one delay but uninformative at neighboring lags. Therefore, fixed input windows may obscure this structure by bundling informative and irrelevant history together. Transfer entropy (TE) can recover these delays by measuring how much a source's past tells us about a target beyond the target's own history. However, existing TE estimators require separate fits across source and conditioning depths, with further estimation on local windows to track delays over time. We introduce ATEMPO, a diffusion-based transfer entropy estimator amortized over lags and time. ATEMPO samples source and conditioning depths during training and conditions explicitly on temporal position, allowing a single score network to estimate the full TE surface without refitting. Differencing this surface recovers when information arrives. Across synthetic datasets with ground-truth TE, ATEMPO achieves the lowest mean relative error among baselines, and on datasets with known lags, ATEMPO achieves 0.98 ± 0.04 lag-detection accuracy. ATEMPO requires a single fit across source depths $K = 6$ and target depths $L = 3$, whereas existing estimators require 18 separate fits to evaluate the same $K \times L$ grid. On three systems with time-varying lag, ATEMPO is the only estimator to maintain perfect lag-detection accuracy across all evaluated window lengths. These recovered delays improve downstream forecasting: ATEMPO-selected inputs reduce MSE on WEATHER-5K by 7.25%, while 120 selected inputs outperform the benchmark's 240-feature default by 4.6%.

1 INTRODUCTION

Multivariate time-series models must decide which variables and past observations to expose to the model. In practice, this is often handled by applying a fixed-width window uniformly to every variable, selecting its length and offset using a validation sweep (Nie et al., 2023; Liu et al., 2024; Han et al., 2026). Yet a variable may become informative only at a particular delay, the gap between when it is observed and when its influence arrives. If that delay is thirty steps, a window long enough to reach it also includes the twenty-nine intervening lags, most of which carry nothing. Although an ideal model could learn to ignore irrelevant history, with finite data these additional inputs increase estimation difficulty and can obscure the value of the delayed signal. This problem compounds in high-dimensional settings, where long common windows expose many potentially uninformative lags. Window selection therefore conflates two questions: *whether* a variable is informative and *when* its information arrives. Identifying the delay directly would allow a model to receive the informative observations without requiring all of the history in between. Transfer entropy (TE) (Schreiber, 2000) provides a natural way to identify such delays: it measures whether a source's past provides information about a target beyond that already contained in the target's own history.

The challenge is that delay identification requires more than one TE estimate, and repeated fitting is costly. Both the source depth k and conditioning depth l are unknown, and changing relationships must also be resolved over time t. The target is therefore a surface $TE(k, l, t)$, where existing estimators typically resolve individual points on this surface. Tracking changing delays compounds the problem, as smaller local windows give better temporal resolution but fewer samples from which to estimate TE. Nearest-neighbor methods need exponentially many samples as the conditioning dimension grows (Gao et al., 2015); variational bounds can become loose at large mutual infor-

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

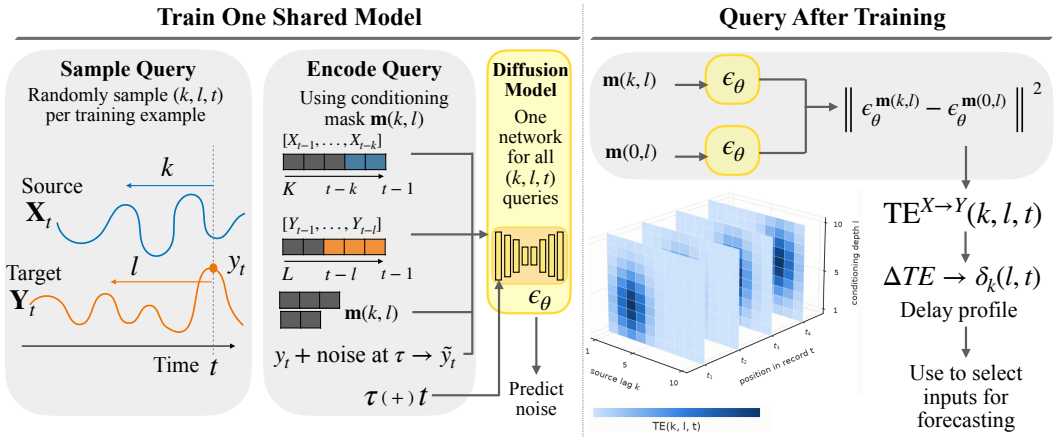

Figure 1: Overview of ATEMPO: During training, source depth k, conditioning depth l and time position t are varied across training samples. A mask $\mathbf{m}(k, l)$ specifies which source and target lags are observed, while t and diffusion time τ conditions through the same pathway. After training, the same network is evaluated with and without the source history; the squared difference between the resulting conditional scores estimates $TE(k, l, t)$. Querying different (k, l, t) recovers the TE surface without refitting, and differencing along k yields the time-varying delay profile. Using these lags as inputs to a forecasting model improves performance over consecutive-timestep baselines.

mation(Belghazi et al., 2018; McAllester & Stratos, 2020); and cross-entropy estimators escape that bound but must discretize the continuous series (Mukherjee et al., 2020; Shalev et al., 2024). Diffusion-based estimators (Munoz et al., 2026) avoid these restrictions, however, they still require repeated fits as source and conditioning depths must be fixed before training.

Our key insight is that the estimators required across this surface differ in their conditioning information, not in the underlying estimation problem. We therefore condition a shared score model on source depth, target-conditioning depth, and temporal position, sampling these configurations during training. The resulting estimator, **ATEMPO** (Amortized Transfer Entropy Model Per-lag Over-time), learns the full TE surface in a single fit, turning quantities previously fixed during training into inference-time choices. Our contributions are:

1. **Amortization over lags and over time.** ATEMPO amortizes estimation across source depth, conditioning depth, and temporal position, replacing separate fits across the TE surface with one conditional score model. It achieves the lowest mean relative error $8.3 \pm 3.3\%$ across synthetic systems in a single fit, whilst others require 18 fits. For time-varying systems, it maintains perfect lag-detection accuracy across all window lengths.
2. **Recovered delays provide useful inputs for downstream prediction.** Identifying where predictive information lies in a time series can provide more useful inputs than exposing a forecasting model to more history. On WEATHER-5K, delays selected using only 300 stations reduce iTransformer MSE by 7.25% across all $5{,}672$ stations; using only half the benchmark's default input features still improves MSE by 4.6%.

2 Related Work

Lag selection with transfer entropy. Choosing the lag badly is not a minor error: it inflates the estimate and produces spurious detections (Faes et al., 2011). Much of the literature is therefore methods for identifying delayed dependence. Granger causality, which coincides with transfer entropy for linear Gaussian processes, is confined to linear couplings. Time-delayed mutual information (Fraser & Swinney, 1986) handles nonlinear ones and recovers the direction of interaction where Granger causality fails (Li et al., 2018), but cannot separate what the driving series contributes from what both series share through a common input. Conditioning on the target's own past resolves

this, and is what transfer entropy does (Schreiber, 2000). Built on it, momentary information transfer (Pompe & Runge, 2011) and $\mathrm{TE}_{\mathrm{SPO}}$ (Wibral et al., 2013) compare couplings across candidate delays, while non-uniform embedding (Faes et al., 2011) admits lags one at a time by a greedy information criterion. Graph-based methods instead search dependencies over a maximum lag (Runge, 2020; Pamfil et al., 2020). A shared limitation is that when each candidate lag is estimated on its own, their errors are independent. This means they do not cancel when you difference TE estimates across lags, and that comparison is needed to identify the lag contributing the most information.

Time-varying coupling. Time-resolved TE is typically obtained through local or sliding-window estimation, trading temporal resolution against the samples available within each window. Related work on nonstationary causal discovery instead incorporates temporal information directly, for example CD-NOD (Huang et al., 2020) introduces a time/domain index to represent changing distributions. ATEMPO combines these ideas differently: temporal position conditions a shared TE estimator trained over the complete recording, while a local window is used only to aggregate its estimates at evaluation. Window width is therefore decoupled from the samples available for fitting the score model.

Estimating transfer entropy. Classical TE estimators inherit the difficulties of conditional mutual-information estimation. Nearest-neighbor methods need exponentially many samples as the conditioning set grows (Gao et al., 2015). Variational lower bounds, built on generative models (Mondal et al., 2020), discriminative critics (Belghazi et al., 2018) or transformers (Luxembourg et al., 2025), can become loose at large information values (McAllester & Stratos, 2020). Cross-entropy estimators escape that bound by taking a difference of entropies, as in CCMI (Mukherjee et al., 2020) and NJEE (Shalev et al., 2024), but discretize continuous data to do it. Diffusion-based estimators provide an alternative by expressing information quantities through differences of learned score functions. MINDE (Franzese et al., 2024) introduced this approach for mutual information, and SΩI (Bounoua et al., 2024) showed that multiple score configurations required by a multivariate information quantity can be amortized within a single denoising model. TENDE (Munoz et al., 2026) applies conditional diffusion scores specifically to transfer entropy and is the closest estimator to ours. However, TENDE fixes the source and target history configuration and window size before training, so the cost remains one fit per hypothesis and per window.

No method therefore varies all three axes. ATEMPO instead amortizes the TE query itself: (k, l) specify the conditioning set and t the temporal position, allowing one model to recover $TE(k, l, t)$ without refitting.Section 3 states the estimation problem and Section 4 describes the model.

3 Problem setting

We seek to identify when information from a source X reaches a target Y, after accounting for information already contained in the target's own past. To do this, we observe one recording $\{(x_t, y_t)\}_{t=1}^{T}$ of a bivariate process with $X_t \in \mathbb{R}^{N_x}$ the source and $Y_t \in \mathbb{R}^{N_y}$ the target. Write $\mathbf{X}_t^{(k)} := (X_{t-1}, \ldots, X_{t-k})$ and $\mathbf{Y}_t^{(l)} := (Y_{t-1}, \ldots, Y_{t-l})$ for histories of depth k and l, empty at depth zero, and K, L for the deepest considered. Transfer entropy (Schreiber, 2000) is the conditional mutual information

$$\mathrm{TE}^{X\to Y}(k,l) \;:=\; I\Big(Y_t;\, \mathbf{X}_t^{(k)} \;\Big|\; \mathbf{Y}_t^{(l)}\Big) \;=\; \mathbb{E}\left[\log \frac{p\big(Y_t \mid \mathbf{X}_t^{(k)}, \mathbf{Y}_t^{(l)}\big)}{p\big(Y_t \mid \mathbf{Y}_t^{(l)}\big)}\right], \tag{1}$$

zero exactly when the source history adds nothing beyond the target's own past. Exchanging the roles of the two series gives $\mathrm{TE}^{Y\to X}$. Conditional mutual information is symmetric in its first two arguments, so the asymmetry comes not from the measure but from which series supplies the history.

Locating the delay. A single evaluation of Eq. 1 scores the block $\mathbf{X}_t^{(k)}$ as a whole: it says the last k source samples jointly inform the target, not which of them does. Differencing in k resolves the block, since $\mathbf{X}_t^{(k)} = (\mathbf{X}_t^{(k-1)}, X_{t-k})$ and the chain rule gives, for $k \geq 1$,

$$\delta_k(l) \;:=\; \mathrm{TE}^{X\to Y}(k,l) - \mathrm{TE}^{X\to Y}(k-1,l) \;=\; I\Big(Y_t;\, X_{t-k} \;\Big|\; \mathbf{X}_t^{(k-1)}, \mathbf{Y}_t^{(l)}\Big) \;\geq\; 0. \tag{2}$$

The vector $\boldsymbol{\delta}(l) = (\delta_1(l), \ldots, \delta_K(l))$ is the *delay profile*: a large entry at k means the source informs the target k steps later, and a run of large entries means the influence persists beyond information provided by earlier lags. By differencing adjacent values, Eq. 2 decomposes $\mathrm{TE}^{X\to Y}(k, l)$ from a cumulative measure of information in the first k source lags to a profile over individual delays. Neither scoring delays in isolation nor conditioning on all others has this property; Appendix C compares them.

The delay profile resolves information across source lags, but it still assumes that two quantities are known. First, its value depends on the target-history depth (l): too little history may attribute to the source information already contained in the target's past, whilst unnecessarily deep conditioning increases the dimension of the estimation problem. We therefore require the profile across both source depth k and conditioning depth l. A long recording adds a third axis if we don't want to be constrained to lags fixed across time. We therefore seek not a single TE estimate, but a surface over source depth, conditioning depth and time: $\mathrm{TE}(k, l, t)$ for Eq. 1 with the law taken over a window of width w centred at t, and $\mathcal{I}$ for the admissible centres. Recovering this surface yields the delay profile at every conditioning depth and position by differencing along its first axis.

Problem (Transfer entropy over lags and time)**.** Given a single recording of two series, determine how much the past of each informs the present of the other at every combination of three quantities: the delay $k \le K$ at which the source is read, the depth $l \le L$ of the target's own past that is held fixed, and the position $t \in \mathcal{I}$ in the recording at which the estimate is taken. Formally, recover the surface

$$\left\{ \mathrm{TE}^{X\to Y}(k, l, t),\ \mathrm{TE}^{Y\to X}(k, l, t) \right\}, \qquad (k, l, t) \in \{1, \ldots, K\} \times \{1, \ldots, L\} \times \mathcal{I}, \tag{3}$$

without fitting a separate estimator for every point on that surface.

Computational and statistical cost. One estimator per cell of Eq. 3 costs $2(K{+}1)(L{+}1)|\mathcal{I}|$ fits. The conditioning set has dimension $kN_x + lN_y$, growing along the very axes being scanned, while a window of width w supplies at most w rows for every cell it contains. Widening the window buys samples at the cost of the time resolution the third axis exists to provide.

4 Method

To estimate transfer entropy across source depth k, conditioning depth l and position t, existing estimators would require a separate fit at each cell, corresponding to every hypothesis about where and when information flows. ATEMPO avoids these repeated fits through a single principle: the quantities that specify the TE hypothesis are inputs to the estimator, rather than properties fixed during training. Diffusion-based TE estimation is particularly suited to this construction. Unlike estimators that require discretization or explicit density estimation, it operates directly on continuous variables and accommodates nonlinear dependence. More importantly, it expresses transfer entropy through two score functions that differ only in their conditioning information. We exploit this structure by training one score model across conditioning configurations and temporal positions. Source-history depth and target-history depth are specified by a conditioning mask, while position enters through the same conditioning path as diffusion time. After training, changing (k, l, t) changes the model's conditioning rather than its parameters. An overview of our method is shown in Figure 1.

4.1 Transfer entropy from score functions

The log-ratio in Eq. 1 is already a Kullback–Leibler divergence between two conditional distributions of the target. Taking the inner expectation over Y_t and the outer over the histories, transfer entropy is expected divergence between these distributions, which differ only in whether the source history is supplied. ATEMPO is based on the principle that if this divergence can be estimated by a model whose conditioning can be varied, then k and l need not be fixed when the model is trained.

Diffusion-based estimators express the divergence without explicitly estimating the density, but instead through two score functions, obtained by regression. Let a variance-preserving SDE (Song et al., 2021) with schedule $\beta(\tau) > 0$ act on the target alone for $\tau \in [0, \bar{\tau}]$, giving $u_\tau = \kappa_\tau u_0 + \sigma_\tau \varepsilon$

with $\kappa_\tau = \exp(-\frac{1}{2}\int_0^\tau \beta(\tau))$, $\sigma_\tau^2 = 1 - \kappa_\tau^2$ and $\varepsilon \sim \mathcal{N}(0, I)$, so the transition kernel is closed-form and no trajectory is simulated. The histories enter only as conditioning. Writing $p_\tau(\cdot \mid c)$ for the law of u_τ when $u_0 \sim p(\cdot \mid c)$, Franzese et al. (2024) give

$$\mathbb{E}_c\big[D_{\mathrm{KL}}(p(\cdot \mid c) \,\|\, q(\cdot \mid c))\big] = \int_0^{\bar\tau} \frac{\beta(\tau)}{2}\, \mathbb{E}_c\, \mathbb{E}_{u\sim p_\tau(\cdot|c)} \|\nabla_u \log p_\tau(u \mid c) - \nabla_u \log q_\tau(u \mid c)\|^2 \,\mathrm{d}\tau, \tag{4}$$

up to a residual at $\tau = \bar\tau$ exponentially small in $\int_0^{\bar\tau} \beta(\tau)$, since both diffused laws converge to $\mathcal{N}(0, I)$. The conditional score is fixed by the posterior mean of the noise, $\nabla_u \log p_\tau(u \mid c) = -\,\mathbb{E}[\varepsilon \mid u_\tau = u, c]/\sigma_\tau$, which a network $\epsilon_\theta(u, \tau, c)$ attains at the optimum of the denoising objective (Vincent, 2011; Ho et al., 2020). Substituting both scores into Eq. 4 and drawing τ from $q(\tau) \propto \beta(\tau)/\sigma_\tau^2$ with normaliser $Z = \int \beta\sigma^{-2}\mathrm{d}\tau$ gives

$$\mathrm{TE}^{X\to Y}(k, l) \;=\; \frac{Z}{2}\, \mathbb{E}_{\tau\sim q,\, \varepsilon} \left\|\epsilon_\theta\big(u_\tau, \tau, (\mathbf{x}, \mathbf{z})\big) - \epsilon_\theta\big(u_\tau, \tau, \mathbf{z}\big)\right\|^2, \tag{5}$$

so one cell costs two forward passes under different conditioning. TENDE (Munoz et al., 2026) evaluates Eq. 5 at a fixed (k, l), with the conditioning structure built into the network's input layer before training. Therefore returning the full $TE(k, l, t)$ surface would require a fit per cell.

Our key observation is that changing (k, l) does not change the score-estimation problem, only which variables the score is conditioned on. We therefore make the conditioning structure itself an input to a shared score model.

4.2 Amortizing over the transfer-entropy surface

ATEMPO trains one score network across the full (k, l, t) surface. To vary (k, l) without refitting, we provide the maximum histories and use a mask to encode which lags are conditioned on:

$$\mathbf{m}(k, l) \;=\; \big[m_y,\; m_{x,1}, \ldots, m_{x,K},\; m_{z,1}, \ldots, m_{z,L}\big] \in \{-1, 0, 1\}^{1+K+L}, \tag{6}$$

where $m_y = 1$ marks the variable whose score is learned, 0 a variable supplied as conditioning, and -1 a variable withheld. For cell (k, l), the first k source lags and l target lags are conditioned on and all deeper lags are withheld. Withheld values are zeroed before projection, while the mask $\mathbf{m}$ is also supplied to the network to distinguish them from observed zeros. Thus, changing (k, l) changes only the mask, not the network.

Training objective. Combining denoising score matching with the cell drawn per example,

$$\mathcal{L}(\theta) \;=\; \mathbb{E}_{\substack{(k,l)\sim\mathcal{U},\; t\sim\mathcal{I} \\ \tau\sim q,\; \varepsilon\sim\mathcal{N}(0,I)}} \Big\| \epsilon_\theta\big(\underbrace{\kappa_\tau y_t + \sigma_\tau\varepsilon,\; \tau}_{\substack{\text{noised target,} \\ \text{diffusion time}}},\quad \underbrace{t,\; \mathbf{m}(k,l)}_{\substack{\text{sampled time,} \\ \text{mask of sampled depths}}},\quad \underbrace{\mathbf{X}_t^{(K)}, \mathbf{Y}_t^{(L)}}_{\substack{\text{maximum histories,} \\ \text{masked by } \mathbf{m}}}\big) - \varepsilon \Big\|^2 \tag{7}$$

with $(k, l) \sim \mathcal{U}\{0, \ldots, K\} \times \mathcal{U}\{0, \ldots, L\}$. Eq. 7 differs from the objective of Munoz et al. (2026) in that: (k, l) is drawn per example instead of fixed before training, and the position t is supplied as a conditioning argument. The forward process, the denoising target and the estimator are unchanged. That single change replaces the $(K{+}1)(L{+}1)$ fits per window of Eq. 3 with one fit in total.

Conditioning on temporal position. The mask amortizes over (k, l), but detecting a lag that changes over the recording additionally requires the score to depend on position t. Rather than fitting a separate model in each time window, we condition the shared score network on t. Diffusion models already condition on diffusion time τ, which is sinusoidally embedded and injected into every residual block as a scale and shift. We reuse this pathway for recording position, normalizing t to $[0, 1]$ and embedding it on the same basis with a separate frequency scale s_t. The embeddings are summed, $h = h_\tau + h_t$, before the modulation

$$h^{(b+1)} \;=\; h^{(b)} + f_b\Big(\mathrm{GroupNorm}(h^{(b)}) \odot (1 + \gamma_b) + \beta_b\Big), \qquad (\gamma_b, \beta_b) = W_b h. \tag{8}$$

Routing t through this pathway lets position information, like diffusion time, modulate features throughout the network rather than enter only at the input projection. Thus τ identifies where in the diffusion process the sample lies, while t tells it where it lies in the recording.

After one fit, any point on the TE surface is recovered by Eq. 5 with the two conditionings addressed by the encoding,

$$\mathrm{TE}^{X\to Y}(k,l,t) \;=\; \frac{Z}{2}\,\mathbb{E}_{\tau\sim q,\;(y,\mathbf{x},\mathbf{z})\in\mathcal{W}_t,\;\varepsilon}\left\|\epsilon_\theta^{\mathbf{m}(k,l)} - \epsilon_\theta^{\mathbf{m}(0,l)}\right\|^2, \tag{9}$$

where $\mathbf{m}(0,l)$ denotes that every source lag is withheld, and the superscript abbreviates the arguments of Eq. 7.

The position t conditions the score network, whereas the window $\mathcal{W}_t$ only determines which local estimates are averaged. We therefore choose its width at evaluation rather than fixing it during training. This temporal sharing assumes that variation in the conditional distribution across the recording can be represented by a shared function of the observations and position t, rather than requiring each local window to define an independent estimation problem. The delay profile Eq. 2 then follows by differencing Eq. 9 in k. Thus, ATEMPO separates training from the hypothesis required for delay discovery: one network learns the conditional scores, the TE surface is queried after training, and differencing that surface locates when source information reaches the target.

5 Experiments

Datasets. WEATHER-5K (Han et al., 2026) contains hourly observations of five meteorological variables from 5,672 stations over 2014-2023. We use the benchmark train/validation/test split (2014-2021/2022/2023) and evaluate forecasting on all 5,672 stations and a fixed 300-station subset. ATEMPO is fit on the 300 station's training data, once for each of the 25 ordered variable pairs ($K = 336$h, $L = 24$h), and the resulting lags are shared across stations. This tests whether delay structure learnt from a small subset remains useful at full-dataset scale. We use iTransformer as the downstream forecaster because it supports the non-consecutive inputs selected by ATEMPO.

To compare baselines for exact TE estimates and lag detection, we use seven synthetic datasets with ground truth TE values; an additional five time-varying datasets makes 12 with ground truth lags. Experiments across window sizes are performed on three of the datasets where the source-to-target delay switches from lag at $k = 2$ to $k = 5$ halfway through an 8,000 sample recording. These three datasets cover linear, mean-blind squared and effect-size-matched squared coupling. Further dataset details are in Appendix A. To extend beyond time series data, we use MMTT-Bench (Andrews & Mengaldo, 2026), which pairs a synthetic signal with natural language descriptions of what the signal is about to do. Full experimental protocols are detailed in Appendix B.1.

Baselines. We compare our method to eight baselines, all of which require a model trained for each (k,l) pair: TENDE (Munoz et al., 2026) uses a diffusion architecture; NPEET computes CMI from kNN statistics; neural estimators NJEE and MINE calculate TE as the difference between entropies; CCMI estimates CMI directly using a binary classifier; and PCMI computes CMI on symbolic encodings of the values rather than the values themselves. To extend our comparison for lag detection we include baselines Liang-Kleeman (Liang, 2013) and DTW, which do not estimate transfer entropy in nats directly, and are therefore excluded from evaluations of explicit TE values. Full implementation details can be found in Appendix B.2.

5.1 Results

WEATHER-5K In a single fit, ATEMPO ranks n features from the preceding 336 hours. Despite selecting these lags using only 300 stations, the resulting inputs reduce forecasting MSE by 7.25% relative to the WEATHER5K default across all 5,672 stations. This shows that the selected lag structure transfers beyond the stations on which it was estimated. The selected lags also exhibit structure consistent with known meteorological periodicities: prominent delays occur at 24-hour intervals, as expected from the diurnal cycle (Dai & Trenberth, 2004; Wu et al., 2021) and further examined in Appendix F. On the 300-station subset, Table 1 shows that ATEMPO-selected inputs reduce forecasting MSE by 6.10% using 240 features and 4.6% using only 120, half the features used by benchmark default we are comparing with. CCMI is the only baseline to achieve comparable forecasting performance, but cannot amortize estimation across lags: with conditioning depth fixed,

[1]Liang-Kleeman and DTW do not predict entropy directly and are excluded from direct TE comparison.

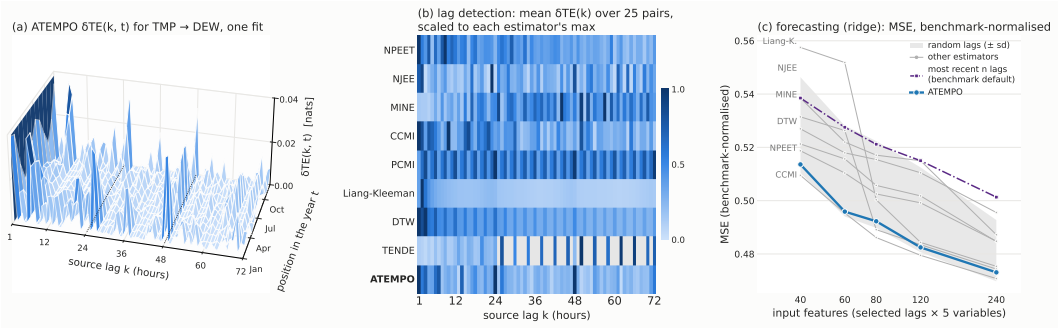

Figure 2: ATEMPO applied to WEATHER5K. a) A delay profile between source variable temperature (TMP) and target variable dew point temperature (DEW) over time, pooled over 300 stations b) The mean delay profile $\delta_k = TE(k) - TE(k-1)$ averaged over 25 variable pairs for each baseline. The darker colors are the lags carrying the most information. c) Forecasting performance (MSE) from using each method's top ranked n features as inputs.

Table 1: iTransformer forecasting error on WEATHER-5K at the benchmark's input size (240 features). MSE is averaged across training seeds.

	MSE (↓)	Δ vs default (↓)	selection cost (h) (↓)
(a) all 5,672 stations			
Default	0.5562	-	0
ATEMPO	**0.5158**	-7.25%	1.8
(b) 300 stations			
Default	0.5223 ±0.0007	-	0
Random	0.5066 ±0.0016	-3.00%	0
Liang	0.5039 ±0.0016	-3.51%	0.5
DTW	0.5021 ±0.0007	-3.85%	0.2
NPEET	0.5005 ±0.0004	-4.16%	2.6
PCMI	0.4994 ±0.0020	-4.38%	1.3
MINE	0.5222 ±0.0015	-0.01%	4.6
NJEE	0.4971 ±0.0016	-4.81%	2.9
CCMI	**0.4904 ±0.0009**	-6.10%	4.4
ATEMPO	**0.4904 ±0.0014**	-6.10%	1.8

Table 2: Mean relative error in TE and top-1 lag detection accuracy across all synthetic datasets and baselines[1]

method	mean rel err % (↓)	Lag detection acc. (↑)	MMTT-Bench acc. (↑)
Liang–Kleeman	-	0.42 ± 0.32	0.00
DTW	-	0.69 ± 0.47	0.00
NPEET	56.0± 38.7	0.92 ± 0.12	0.33
PCMI	55.1± 22.1	0.91 ± 0.14	**1.00**
NJEE	32.9± 13.1	0.93 ± 0.11	0.72
CCMI	24.0± 26.4	0.95 ± 0.07	0.78
TENDE	16.3± 14.0	0.95±0.12	0.94
MINE	11.9 ± 8.3	0.97 ± 0.05	0.83
ATEMPO	**8.3 ± 3.3**	**0.98±0.04**	0.94

evaluating $K = 336$ source lags for each of the 25 ordered variable pairs requires 8,400 CCMI fits, compared with ATEMPO's 25. Therefore lag selection costs 4.4 hours for CCMI, 1.8 for ATEMPO.

Synthetic Datasets On synthetic datasets with ground truth TE and known lags, Table 2 shows ATEMPO produces the most accurate transfer entropy estimates and the best lag detection accuracy. It achieves this in a single fit for source depths $K = 6$ and target depths $L = 3$, where other estimators require $K \times L = 18$ separate fits, one for each (k,l) pair. Comparing to the most similar model architecture TENDE, the time per fit is approximately 5.8 minutes for both TENDE and ATEMPO, so amortizing over the lag grid costs nothing in terms of time-per-fit, but produces a cost saving proportional to $K \times L$ overall. Figure 3 demonstrates the value of conditioning on time position on three synthetic datasets where the source-to-target delay changes partway through a recording. Without time conditioning an estimator has two choices. It can refit every window, shown in dark gray, which works while windows are long but decays as they shrink as each fit sees fewer samples - the best baseline estimator falls from 0.92 to 0.68 lag detection accuracy as window size drops from 1000 to 125. Alternatively it can fit once on the whole recording and report a single lag, shown in light gray, but this cannot exceed 0.5 accuracy when the lag switches once within the dataset. ATEMPO is fitted once on the same 8,000 samples and recovers the lag in every window at every length. The margin over the best refit baseline grows from 8-32% as windows shrink.

Using the same three datasets, we quantify the extent to which accurate lag detection improves downstream forecasting. Figure 4 shows the correlation between detection rate and MSE improve-

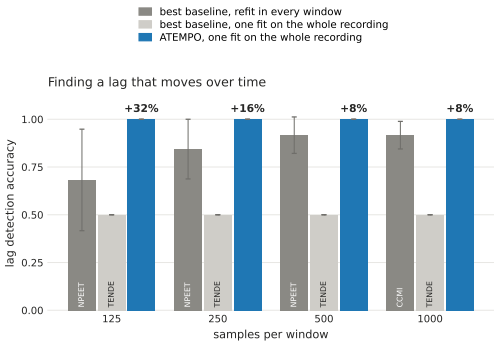

Figure 3: Lag detection accuracy against window length, mean and std over three time-varying systems. Each method reports one lag per window; accuracy is the fraction of windows whose top-1 lag is the true one.

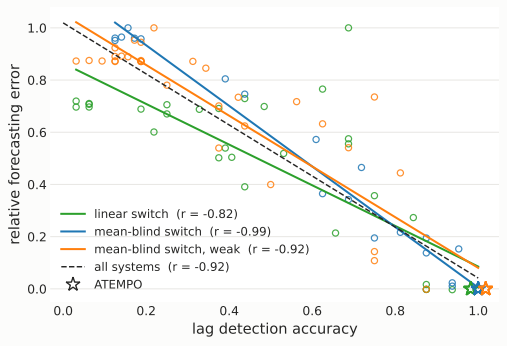

Figure 4: Correlation between lag detection accuracy and forecasting error for three time-varying systems. Relative forecasting error is the MSE error compared to the MSE when using true lags for forecasting.

ment, where each point is one estimator at one window length. A least-squares fit per dataset gives the Pearson correlations $r = [-0.82, -0.99, -0.92]$ per system ($n = 36$, two-sided t-test on r gives nominal p between 10^{-9} and 10^{-31}). Full results can be found in Appendix D.

MMTT-Bench The value of utilizing information driving time series is not restricted to a single modality. MMTT-Bench (Andrews & Mengaldo, 2026) pairs a synthetic signal with natural language descriptions of what the signal is about to do. Natively the text description at step t predicts the time series at $t+1$, a lag of one. Re-aligning the two streams plants an arbitrary delay m, giving a controlled ground truth in a multimodal forecasting setting. The benchmark supplies correct and irrelevant descriptions per timestep, where irrelevant text adds no additional information to the time series forecasting task. Our estimator recovers the planted lag perfectly for five out of six lags and identified zero lags in the null irrelevant category, equaling TENDE and beating all other baselines except for PCMI in comparable processing time. Furthermore ATEMPO correctly estimates TE values highest for the correct text compared to irrelevant (0.52 to 0.14 nats), and this corresponds to an improvement in forecasting MSE of 41% for correct. Full results are available in Appendix E.

Ablation. Table 3 separates two benefits of ATEMPO: amortizing over lag and conditioning on temporal position. When both models are fit once to the complete recording, amortizing over lag alone detects the moving lag in only 0.58-0.64 of windows. Adding temporal conditioning raises detection to 1.00 at every evaluation-window length. Thus, access to the full recording alone is insufficient: the estimator must also represent how the dependence changes over time.

Fitting estimators independently within each window has the opposite limitation. These models can represent local dependence, but their accuracy deteriorates as fewer samples are available for each fit. NPEET, the strongest baseline at 125 samples, falls from 0.92 detection at 1,000 samples to 0.68 at 125; ATEMPO without time conditioning falls from 1.00 to 0.06. Full ATEMPO avoids this trade-off: its time-conditioned score model is trained once on the complete recording, while the evaluation window determines only where TE estimates are aggregated. It therefore retains the statistical strength of the full recording while resolving local changes in lag, maintaining 1.00 detection with 125-sample evaluation windows.

These differences propagate directly to forecasting. We report MSE relative to a forecaster given the known lag in every window, so 1.0 indicates oracle-level performance. As per-window lag detection deteriorates, forecasting error generally increases. Conversely, estimators fit once without temporal conditioning select a global lag and incur approximately $4\times$ the oracle error despite using the entire recording. Full ATEMPO recovers the local lag in every window and consequently matches the oracle forecasting performance at every window length.

Limitations. ATEMPO amortizes transfer entropy estimation across lag, conditioning depth and time, but does not remove the statistical requirements of estimating conditional dependence. In particular, at large dimensionality, specified by (K, l), there is a minimum requirement for data

Table 3: Lag detection and forecasting under a time-varying lag. Results are averaged over three synthetic systems. Lag detection accuracy is the fraction of evaluation windows in which the top-ranked lag equals the true lag. Relative forecasting error is the MSE error compared to the MSE when using true lags for forecasting. Per-window methods are refit independently using only the samples in each window. Fit-once methods are trained on the complete 8,000 sample recording. Only time-conditioned ATEMPO can vary its estimated lag across windows without refitting.

	lag detection accuracy (↑)				Relative forecasting error (↓)			
samples per window	125	250	500	1000	125	250	500	1000
fitted per window								
NPEET	0.68	0.84	0.92	0.92	1.83	1.09	1.05	1.01
NJEE	0.35	0.44	0.62	0.88	2.80	1.83	1.17	1.09
MINE	0.32	0.52	0.67	0.92	3.87	1.28	1.17	1.02
CCMI	0.19	0.42	0.54	0.92	4.70	2.10	1.16	**1.00**
PCMI	0.29	0.48	0.67	0.75	4.98	2.21	1.21	1.08
Liang-Kleeman	0.23	0.20	0.23	0.25	5.69	5.68	5.64	5.70
DTW	0.43	0.48	0.73	0.71	2.90	3.35	1.11	1.13
TENDE	0.11	0.11	0.35	0.88	5.73	5.90	2.10	1.07
ATEMPO (no time conditioning)	0.06	0.12	0.62	**1.00**	5.73	5.72	1.91	**1.00**
fitted once on the whole recording								
NPEET	0.50	0.50	0.50	0.50	4.50	4.50	4.50	4.50
NJEE	0.50	0.50	0.50	0.50	4.00	4.00	4.00	4.00
MINE	0.50	0.50	0.50	0.50	4.01	4.01	4.01	4.01
CCMI	0.50	0.50	0.50	0.50	4.00	4.00	4.00	4.00
PCMI	0.50	0.50	0.50	0.50	4.00	4.00	4.00	4.00
Liang-Kleeman	0.17	0.17	0.17	0.17	5.71	5.71	5.71	5.71
DTW	0.50	0.50	0.50	0.50	4.49	4.49	4.49	4.49
TENDE	0.50	0.50	0.50	0.50	4.01	4.01	4.01	4.01
ATEMPO (no time conditioning)	0.64	0.60	0.60	0.58	3.96	4.01	4.00	4.01
ATEMPO (time-conditioned)	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**

samples and model capacity. A second limitation concerns the recovered delay profile, as errors in the transfer entropy estimation can propagate through. Detailed analysis in Appendix F.1 shows that not all selected lags add equal value to downstream forecasting, implying residual errors in TE estimation can lead to less-than-optimal lag selection. Finally the scope of our empirical evaluation is necessarily limited. Exact transfer entropy is available only for controlled processes, so quantitative estimation error is evaluated on synthetic systems which are approximately Gaussian and contain a single informative dimension. We complement these experiments with MMTT-Bench and WEATHER5K, where ground truth TE is unavailable and utility is instead assessed through known alignment or downstream forecasting.

Future Work. The central idea of ATEMPO, turning quantities normally fixed during estimation into conditioning variables of a shared model, could be extended beyond pairwise transfer entropy to condition jointly on multiple sources. This would allow the estimator to distinguish information that is unique to one variable from information that is redundant or synergistic across variables, turning the present lag-selection procedure to joint variable-and-lag selection for high-dimensional forecasting.

6 Conclusion

Fixed input widths conflate two questions: how much history to expose to a model, and where within that history useful information actually lies. We introduced ATEMPO to separate them. By expressing transfer entropy through conditional diffusion scores and making lag, conditioning depth and temporal position inputs to a shared estimator, ATEMPO replaces separately fitted models with a single amortized fit. The resulting $TE(k, l, t)$ surface can be queried across histories and time, and differenced along k to recover a time-varying delay profile. We show ATEMPO achieves the best mean TE estimations and lag-detection accuracy compared to 8 baselines. These recovered delays then reduce forecasting MSE by 5-7% across the WEATHER5K dataset.

AI USE STATEMENT

In this work, we used generative AI tools to check our implementations of TE estimator baselines which do not provide open-source code, and review our synthetic dataset generation. We did not use AI to help with any of the mathematical claims or proofs, propose or refine hypothesis or interpret results.

Additionally, we used generative AI to create some of the diagrams used in Figure 1 and generate the code to improve formatting in Figure 2. We also used it clean up and add comments to our code base to make it usable for others, and provide feedback on our research paper draft, specifically in terms of clarity and sentence structure. We have reviewed all AI-assisted work, all manuscript suggestions were made offline and only humans made changes directly to the text after AI feedback. All LLM-generated code was verified and tested for correctness using automated tests, and an author reviewed every AI-generated doc-string. We take full responsibility for the final content of this work.

ETHICS STATEMENT

This work develops a general-purpose method for estimating transfer entropy and identifying time-varying delayed dependencies in sequential data. Our experiments use synthetic data and existing public benchmarks, and do not involve the collection of personal data or human-subject experimentation.

The potential benefits and risks of ATEMPO depend on its application and on the data to which it is applied. While identifying informative delays may improve forecasting and provide insight into temporal dependencies, the same methodology could be applied to sensitive data. Users should consider appropriate privacy, fairness and domain-specific requirements in such settings. We do not advocate using transfer entropy estimates alone for decision-making in high-stakes applications, instead we recommend human-in-the-loop to mitigate risks in these settings.

REPRODUCIBILITY STATEMENT

Our ATEMPO method, all experiments and all figure and table generation is fully reproducible and made available on GitHub[2]. The README contains clear instructions on how to reproduce all experiments, with one script per dataset. A notebook is available to generate all plots and figures from the saved results. Additionally, we have created a 'run_on_your_data.py' to make it easier for others to try ATEMPO. We use fully open-source datasets, with pre-processing details in Section 5 and links to the data on GitHub. More details about our experimental protocol and baseline estimator implementations can be found in Appendix B.1 and B.2, including information about the parameters for ATEMPO and the forecasting models that are not already stated in Section 5.1.

REFERENCES

Emma Andrews and Gianmarco Mengaldo. When does text inform? benchmarking information-theoretic metrics for multimodal time-series forecasting, 2026. URL https://arxiv.org/abs/2609.11282.

Christoph Bandt and Bernd Pompe. Permutation entropy: a natural complexity measure for time series. *Physical review letters*, 88(17):174102, 2002.

Mohamed Ishmael Belghazi, Aristide Baratin, Sai Rajeshwar, Sherjil Ozair, Yoshua Bengio, Aaron Courville, and Devon Hjelm. Mutual information neural estimation. In *International conference on machine learning*, pp. 531–540. PMLR, 2018.

Donald J Berndt and James Clifford. Using dynamic time warping to find patterns in time series. In *Proceedings of the 3rd international conference on knowledge discovery and data mining*, pp. 359–370, 1994.

Mustapha Bounoua, Giulio Franzese, and Pietro Michiardi. S■i: Score-based o-information estimation. In *ICML 2024, 41st International Conference on Machine Learning*, 2024.

[2] https://anonymous.4open.science/r/ATEMPO-F153/README.md

Aiguo Dai and Kevin E Trenberth. The diurnal cycle and its depiction in the community climate system model. *Journal of climate*, 17(5):930–951, 2004.

Luca Faes, Giandomenico Nollo, and Alberto Porta. Information-based detection of nonlinear Granger causality in multivariate processes via a nonuniform embedding technique. *Physical Review E*, 83(5):051112, 2011. doi: 10.1103/PhysRevE.83.051112.

Giulio Franzese, Mustapha Bounoua, and Pietro Michiardi. Minde: Mutual information neural diffusion estimation. In *International Conference on Learning Representations*, volume 2024, pp. 16685–16716, 2024.

Andrew M. Fraser and Harry L. Swinney. Independent coordinates for strange attractors from mutual information. *Physical Review A*, 33(2):1134–1140, 1986. doi: 10.1103/PhysRevA.33.1134.

Shuyang Gao, Greg Ver Steeg, and Aram Galstyan. Efficient estimation of mutual information for strongly dependent variables. In Guy Lebanon and S. V. N. Vishwanathan (eds.), *Proceedings of the Eighteenth International Conference on Artificial Intelligence and Statistics*, volume 38 of *Proceedings of Machine Learning Research*, pp. 277–286. PMLR, 2015. URL https://proceedings.mlr.press/v38/gao15.html.

Tao Han, Zhibin Wen, Zhenghao Chen, Dazhao Du, Song Guo, and LEI BAI. Benchmarking physics-informed time-series models for operational global station weather forecasting. In *Forty-third International Conference on Machine Learning*, 2026. URL https://openreview.net/forum?id=HTlNhKxcrc.

Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. In *Advances in Neural Information Processing Systems*, volume 33, 2020. URL https://proceedings.neurips.cc/paper_files/paper/2020/hash/4c5bcfec8584af0d967f1ab10179ca4b-Abstract.html.

Biwei Huang, Kun Zhang, Jiji Zhang, Joseph Ramsey, Ruben Sanchez-Romero, Clark Glymour, and Bernhard Schölkopf. Causal discovery from heterogeneous/nonstationary data. *Journal of Machine Learning Research*, 21(89):1–53, 2020.

Eamonn J Keogh and Michael J Pazzani. Derivative dynamic time warping. In *Proceedings of the 2001 SIAM international conference on data mining*, pp. 1–11. SIAM, 2001.

Daniel Kornai, Ricardo Silva, and Nikolaos Nikolaou. Agm-te: Approximate generative model estimator of transfer entropy for causal discovery. In Biwei Huang and Mathias Drton (eds.), *Proceedings of the Fourth Conference on Causal Learning and Reasoning*, volume 275 of *Proceedings of Machine Learning Research*, pp. 947–990. PMLR, 07–09 May 2025. URL https://proceedings.mlr.press/v275/kornai25a.html.

Alexander Kraskov, Harald Stögbauer, and Peter Grassberger. Estimating mutual information. *Physical Review E—Statistical, Nonlinear, and Soft Matter Physics*, 69(6):066138, 2004.

Songting Li, Yanyang Xiao, Douglas Zhou, and David Cai. Causal inference in nonlinear systems: Granger causality versus time-delayed mutual information. *Physical Review E*, 97(5):052216, 2018. doi: 10.1103/PhysRevE.97.052216.

Zhaohui Li, Gaoxiang Ouyang, Duan Li, and Xiaoli Li. Characterization of the causality between spike trains with permutation conditional mutual information. *Physical Review E—Statistical, Nonlinear, and Soft Matter Physics*, 84(2):021929, 2011.

X San Liang. The liang-kleeman information flow: Theory and applications. *Entropy*, 15(1):327–360, 2013.

X San Liang. Unraveling the cause-effect relation between time series. *Physical Review E*, 90(5): 052150, 2014.

Yong Liu, Tengge Hu, Haoran Zhang, Haixu Wu, Shiyu Wang, Lintao Ma, and Mingsheng Long. iTransformer: Inverted transformers are effective for time series forecasting. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=JePfAI8fah.

Omer Luxembourg, Dor Tsur, and Haim Permuter. TREET: Transfer entropy estimation via transformers. *IEEE Access*, 13:126477–126495, 2025. doi: 10.1109/ACCESS.2025.3588009.

David McAllester and Karl Stratos. Formal limitations on the measurement of mutual information. In Silvia Chiappa and Roberto Calandra (eds.), *Proceedings of the Twenty Third International Conference on Artificial Intelligence and Statistics*, volume 108 of *Proceedings of Machine Learning Research*, pp. 875–884. PMLR, 2020. URL https://proceedings.mlr.press/v108/mcallester20a.html.

Arnab Kumar Mondal, Arnab Bhattacharjee, Sudipto Mukherjee, Himanshu Asnani, Sreeram Kannan, and A. P. Prathosh. C-MI-GAN: Estimation of conditional mutual information using minmax formulation. In *Proceedings of the 36th Conference on Uncertainty in Artificial Intelligence*, volume 124 of *Proceedings of Machine Learning Research*, pp. 849–858. PMLR, 2020. URL https://proceedings.mlr.press/v124/mondal20b.html.

Sudipto Mukherjee, Himanshu Asnani, and Sreeram Kannan. Ccmi: Classifier based conditional mutual information estimation. In *Uncertainty in artificial intelligence*, pp. 1083–1093. PMLR, 2020.

Simon Pedro Galeano Munoz, Maurizio Filippone, Giulio Franzese, Mustapha Bounoua, and Pietro Michiardi. TENDE: Transfer entropy neural diffusion estimation. In *The 29th International Conference on Artificial Intelligence and Statistics*, 2026. URL https://openreview.net/forum?id=Cje2pdPusB.

Yuqi Nie, Nam H. Nguyen, Phanwadee Sinthong, and Jayant Kalagnanam. A time series is worth 64 words: Long-term forecasting with transformers. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=Jbdc0vTOcol.

Roxana Pamfil, Nisara Sriwattanaworachai, Shaan Desai, Philip Pilgerstorfer, Konstantinos Georgatzis, Paul Beaumont, and Bryon Aragam. DYNOTEARS: Structure learning from time-series data. In *Proceedings of the Twenty Third International Conference on Artificial Intelligence and Statistics*, volume 108 of *Proceedings of Machine Learning Research*, pp. 1595–1605. PMLR, 2020. URL https://proceedings.mlr.press/v108/pamfil20a.html.

Liam Paninski. Estimation of entropy and mutual information. *Neural computation*, 15(6):1191–1253, 2003.

Bernd Pompe and Jakob Runge. Momentary information transfer as a coupling measure of time series. *Physical Review E*, 83(5):051122, 2011. doi: 10.1103/PhysRevE.83.051122.

Jakob Runge. Conditional independence testing based on a nearest-neighbor estimator of conditional mutual information. In *International conference on artificial intelligence and statistics*, pp. 938–947. Pmlr, 2018.

Jakob Runge. Discovering contemporaneous and lagged causal relations in autocorrelated nonlinear time series datasets. In *Proceedings of the 36th Conference on Uncertainty in Artificial Intelligence*, volume 124 of *Proceedings of Machine Learning Research*, pp. 1388–1397. PMLR, 2020. URL https://proceedings.mlr.press/v124/runge20a.html.

H. Sakoe and S. Chiba. Dynamic programming algorithm optimization for spoken word recognition. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 26(1):43–49, 1978. doi: 10.1109/TASSP.1978.1163055.

Thomas Schreiber. Measuring information transfer. *Physical Review Letters*, 85(2):461–464, 2000. doi: 10.1103/PhysRevLett.85.461.

Yuval Shalev, Amichai Painsky, and Irad Ben-Gal. Neural joint entropy estimation. *IEEE Transactions on Neural Networks and Learning Systems*, 35(4):5488–5500, 2024. doi: 10.1109/TNNLS.2022.3204919.

Yang Song, Jascha Sohl-Dickstein, Diederik P Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=PxTIG12RRHS.

Pascal Vincent. A connection between score matching and denoising autoencoders. *Neural Computation*, 23(7):1661–1674, 2011. doi: 10.1162/NECO_a_00142.

Michael Wibral, Nicolae Pampu, Viola Priesemann, Felix Siebenhühner, Hannes Seiwert, Michael Lindner, Joseph T. Lizier, and Raul Vicente. Measuring information-transfer delays. *PLOS ONE*, 8(2):e55809, 2013. doi: 10.1371/journal.pone.0055809.

Haixu Wu, Jiehui Xu, Jianmin Wang, and Mingsheng Long. Autoformer: Decomposition transformers with auto-correlation for long-term series forecasting. *Advances in neural information processing systems*, 34:22419–22430, 2021.

A Dataset Details

Table 4: Details of our suite of synthetic datasets with ground truth TE

Family	Dataset	Source Dimension	Number of true lags	Linear/ Non-linear	Ground Truth TE	Known Lag(s)
AGM TE (Kornai et al., 2025)	flat_linear	1	1	linear		
	flat_joint	1	1	non-linear	✓	✓
multi-dimensional	lag_linear_m3_d4	5	1	linear	✓	✓
	lag_nonlin_m3_d4	5	1	non-linear	✓	✓
	lag_multisource	5	1	linear	✓	✓
mean-blind	lag_sq_m20	1	1	non-linear	✓	✓
	lag_vol_m20	1	1	non-linear	✓	✓
time-varying lag	switch	1	2 (quick change)	linear	×	✓
	drift	1	4 (gradual change)	linear	×	✓
	null	1	1 (no change)	linear	×	✓
time-varying mean-blind	switch_sq	1	2 (quick change)	non-linear	×	✓
	switch_sq_wk	1	2 (quick change)	non-linear	×	✓

Synthetic Datasets Table 4 lists our suite of synthetic datasets and their properties. The datasets flat linear and flat joint are systems equivalent to Kornai et al. (2025)'s linear-VAR and joint process respectively, and we then constructed two multi-dimensional, two mean-blind, three with time-varying lag and two with both mean-blind and time-varying lag datasets. The seven non-time-varying datasets are stationary, because an exact value of $TE(k, l)$ is only tractable for stationary processes. The additional five datasets are used for lag detection and forecasting comparisons only. Datasets only have one informative dimension, but some have up to four redundant dimensions added. We vary the systems coupling nonlinearity, dimensionality and number of delays, and every system is roughly Gaussian and near-symmetric. We selected these families as they target the known limitations of existing TE estimators, high-dimensionality, non-linearity and lags that vary over time. All code to generate our datasets can be found in our GitHub[3].

B Experimental Protocol

B.1 Datasets

WEATHER5K We retain the original WEATHER-5K split using 2014-2021 for training, 2022 for validation and 2023 exclusively for testing. A fixed random subset of 300 stations is drawn once using the training data and used for all lag selection experiments. ATEMPO is trained separately for each of the $5 \times 5 = 25$ ordered source-target variable pairs, with maximum source-history depth $K = 336$h and target conditioning depth $L = 24$h, pooling windows across the 300 stations. Each fitted model yields the full surface $TE(k, l, t)$ but lag selection reads a single column of it: the marginal gain $\delta_k(l)$ of Eq. 2 at the deepest conditioning depth, $L = 24$h. The per-pair profiles are clipped at zero and normalized to unit sum, so that no high-TE pair dominates, then averaged over the 25 ordered pairs, and the (n) lags with the largest pooled gain form the selected set. For a variable against itself the gain at $l = 24$ is zero for $k \leq 24$ by construction, so sub-daily self-lags enter the ranking through the cross-variable pairs only; the forecaster receives the target's recent history regardless.

Lag selection uses only these fitted models and training data; neither validation nor test observations are used to select lags. The selected source lags are then fixed and used as input features across stations. We evaluate the same selections by training forecasting models on both the 300-station subset and the complete 5,672-station dataset, allowing us to test whether lag structure estimated

[3] https://anonymous.4open.science/r/ATEMPO-F153/README.md

from the subset transfers across the full station population. Forecasting follows the benchmark parameters: iTransformer is trained with early stopping using the validation set, while the ridge penalty is selected on validation performance (ridge parameters are the same as specified in the Synthetic Datasets section below). All reported MSEs are evaluated on the held-out 2023 test set and averaged across the five meteorological variables. We use iTransformer because it performs competitively in the original benchmark while, unlike architectures requiring uniformly spaced consecutive inputs, it can directly consume the non-consecutive lag sets selected by ATEMPO.

Synthetic Datasets We split the data chronologically, with the first 70% of points as train, 15% for validation, to choose model parameters, and the final 15% as test for measuring MSE. The forecaster model is a ridge regression with $\alpha \in \{0.1, 1, 10, 100\}$ plus an MLP regressor with (32,) and (64,32) which uses early stopping. The input feature used for forecasting are the targets own past with a lag of l=3, $y[t-l]$ for $l \in 1, 2, 3$, and the source $x[t-k]$ where k is the detected lag at time t. Each (model,window size) pair is trained and averaged across three seeds, apart from ATEMPO where only one model is trained for all window sizes.

For evaluating performance at different window sizes in Figure 3 and our ablation Table 3, we use three synthetic systems: switch, switch_sq and switch_sq_wk. Each is a single recording of $N = 8{,}000$ samples generated with seed 0, in which the source-to-target delay is $m = 2$ for the first $4{,}000$ samples and $m = 5$ for the second. Each recording is cut into $W \in 8, 16, 32, 64$ non-overlapping windows of $1{,}000$, 500, 250 and 125 samples. For lag selection we pool the gain over conditioning depths,

$$\bar{\delta}_k \;:=\; \frac{1}{L}\sum_{l=1}^{L} \delta_k(l), \qquad \hat{k} \;:=\; \arg\max_{k\in\{1,\dots,K\}} \bar{\delta}_k, \tag{10}$$

so that a lag is credited only for information it adds at every depth of the target's own history; $\bar{\delta}_k \geq 0$ follows from Eq. 2. Every method reports one lag per window, the argmax of the pooled gain $\bar{\delta}_k$ of Eq. 10 with $K = 8$ and $L = 2$, and is scored against the planted lag at the window center. Detection accuracy is the fraction of correct windows, averaged over the three systems; error bars in Figure 3 are the standard deviation across systems.

The three arms differ only in what data a fit sees.

- Refit per window: the method is re-estimated on each window's samples alone, one estimate per (k, l) cell, i.e. $K \cdot L \cdot W$ estimates per recording (a trained network per cell for TENDE, MINE, CCMI and NJEE; a plug-in estimate for NPEET, PCMI, Liang–Kleeman and DTW), or W amortised fits for ATEMPO without the time axis.
- Fit once: the method is estimated once on all $8{,}000$ samples and its single lag is applied to every window; without a time axis this is the only other route a baseline has, and it cannot exceed 0.5 when the lag takes two values for equal durations.
- ATEMPO: one amortized fit on all $8{,}000$ samples with the score network conditioned on series time, evaluated per window by restricting the expectation to that window's rows. All ATEMPO and TENDE fits use 400 epochs.

For forecasting results in Table 3 and Table 10, we re-use the detected lags used to calculate lag detection accuracy in Table 3 and Table 9. For each system we draw three independent realizations of the same process (seeds 500–502), so no detector is scored on data it was fitted on; the lag schedule is identical, so a lag sequence detected on the seed-0 recording applies unchanged. A forecaster predicts y_t from its own three most recent values plus the single source value $x_{t-\hat{k}(t)}$, where $\hat{k}(t)$ is the lag the method detected for the window containing t. Within each $4{,}000$-sample half, the first 70% of rows are used for fitting, the next 15% to choose between ridge regression (five penalties, 10^{-2} to 10^{2}) and a small MLP (one or two hidden layers), and the last 15% for the reported test MSE. The oracle is the identical pipeline given the true lag in every row; we report the ratio of test MSE to oracle test MSE, averaged over seeds and then over systems.

MMTT-Bench We concatenate the benchmark's train, validation and test splits into one series of 2048 steps. The target takes four discrete values $\{-1, -0.5, 0, 1\}$; since density-based estimators require a continuous target, we add Gaussian dither with $\sigma = 0.02$. The benchmark's text at step

t describes step $t + 1$ (a native lag of 1). We plant a delay m that advances the text stream by $m - 1$ steps. Four conditions are run: the benchmark's correct, incorrect and irrelevant text, plus a shuffled null in which the correct text is permuted in time on the source side only, leaving the target's own dynamics intact. Sentences are embedded with the benchmark's default encoder, sentence-transformers/all-distilroberta-v1, reduced by PCA and standardized. Detection and forecasting use different PCA dimensions because their sample budgets differ: $d = 4$ for TE estimation ($K = 8$ lags $\times 4$ dims $= 32$ source features against around 2000 samples) and $d = 16$ for the ridge forecaster (128 features against around 1400 training rows).

ATEMPO is fitted once per (condition, planted lag, seed) with $K = 8$ source lags and $L = 3$ target lags, 400 epochs, and otherwise the defaults of the ridge model used in Synthetic Datasets above. The downstream forecasting model predicts y_t from its own three most recent values plus the 16-dimension text embedding at a window of W consecutive lags. The first 70% of the data is used for fitting, with the final 20% of this for validation to select the ridge penalty. The last 30% of the data is the held out test set for calculating forecasting MSE.

B.2 Baselines

We implement eight TE estimator baselines to compare to ATEMPO. Details of our implementations are outlined below, with full code available in our GitHub[3]. As Liang-Kleeman and DTW do not estimate transfer entropy in nats directly, we do not evaluate them for accuracy against ground truth TE. Instead we report their output matrix as a cumulative sum over source lags, such that when the Δ is taken it corresponds to the per-lag quantity, equivalent to the ΔTE surface itself.

- **NPEET**: Non-parametric k-nearest-neighbor CMI computes CMI from k-NN distance statistics in the join space. We use the NPEET python mutual information estimator implementation based on Kraskov et al. (2004) with $k = 3$ neighbors and convert its base-2 output to nats. It needs no training, making it the cheapest baseline, but it is also non-deterministic. Given no seed, repeated calls on identical data can differ in the third decimal.
- **NJEE**: Neural joint entropy estimation estimates transfer entropy as a difference of two conditional entropies $TE = H(Y \mid Z) - H(Y \mid X, Z)$, each obtained as a neural classifier's cross-entropy, which upper-bounds the conditional entropy and tightens as the classifier improves. We discretize the target into 8 equally-probable levels so each entropy becomes a classification loss, leaving the conditional variables continuous, and train a two-hidden-layer MLP (128 units, Adam 400 steps) per term. Cross-entropy is evaluated on a held-out split, because in-sample cross-entropy underestimates entropy and would inflate every TE, and because TE is a difference of two terms the leading discretization bias cancels. For multivariate targets, we only discretize the first target dimension, making its value a lower bound on the true TE.
- **MINE**: The Donsker-Varadhan neural mutual information estimator (Belghazi et al., 2018) where a critic T_θ is trained to maximize $I(A; B) \geq \mathbb{E}p(a, b)[T] - \log \mathbb{E}p(a)p(b)[e^T]$. We use a two-hidden-layer ReLU MLP (128 units) trained with Adam (learning rate 10^{-3}, 400 steps, batch 256), drawing marginal samples by shuffling (B) within each batch, and evaluate the bound on the full sample at the full sample at the final step. Because MINE estimates an unconditional MI, transfer entropy is obtained as a difference of two separately estimated quantities: ($TE = I(Y; X, Z) - I(Y; Z)$). The errors therefore add rather than cancel, which is the decomposition that the next baseline, CCMI, was introduced to avoid.
- **CCMI**: The classifier-based conditional mutual information estimator (CCMI) (Mukherjee et al., 2020) estimates $I(Y; X \mid Z)$ directly. A binary classifier is trained to separate the joint $p(x, y, z)$ from the conditional, producing $p(x \mid z)p(y \mid z)p(z)$, and the Donsker-Varadhan functional applied to its logits, which are pointwise log-density ratio estimates, gives the CMI without the difference of large terms. We construct the conditional-product sample by the local permutation scheme of (Runge, 2018): for each i, x_i is replaced by x_j drawn from among i's five nearest neighbors in Z. A global shuffle would destroy the $X - Z$ relationship and inflate the estimate. Classifier and optimizer settings are the same as MINE.
- **Liang-Kleeman**: Liang-Kleeman information flow is a causality measure derived from the rate of change of the target's marginal entropy once its own self-influence is removed (Liang, 2013). We implement the maximum-likelihood estimator of Liang (2014) in its multivariate form: for the linear stochastic system $\dot{x}i = \sum_j aijx_j + \varepsilon$, the coefficients are $\hat{a} = C^{-1}C_d$ with C the

state covariance and $C_d[k,i] = \mathrm{Cov}(x_k, \dot{x}i)$ using an Euler forward difference, and the flow is $Tj \to i = a_{ij}C_{ij}/C_{ii}$. At two components this reduces algebraically to the published bivariate closed form $T_{2\to1} = (C_{11}C_{12}C_{2,d1} - C_{12}^2C_{1,d1})/(C_{11}^2C_{22} - C_{11}C_{12}^2)$, which we verify numerically. Because the forward difference makes a source sample at t act on the target at $t+1$, probing delay k shifts the source by $k-1$. The final result is a rate in nats per unit time, which is not an estimator of transfer entropy directly, , so it is only judged on lag discovery and downstream forecasting. It also has no target-history-depth parameter, so its matrix is constant along l.

- **DTW**: Dynamic time warping is a lead-lag heuristic rather than an information measure (Berndt & Clifford, 1994). We align the two series with a Sakoe–Chiba band of half-width 64 (Sakoe & Chiba, 1978) and read the offset $j - i$ of every matched pair as a local estimate of how far the target trails the source. The histogram of these offsets over lags $1..K$, as a fraction of matched pairs, is the per-lag profile, and its mode is the detected delay. Both series are z-scored and first-differenced before alignment, following the derivative variant of Keogh & Pazzani (2001): raw DTW is biased by exactly one lag on autoregressive targets, because the target's own level memory drags the alignment forward a step, while differencing leaves the innovation the coupling acts on (top-1 accuracy on the synthetic suite rises from 0.66 to 0.98). The band is fixed rather than tied to K so that every cell of the matrix reads the same alignment and only the histogram window moves. The output is dimensionless and not a bound on TE, so DTW is excluded from the TE error comparison; and because DTW matches shapes without conditioning on the target's own past, it has no l axis and cannot separate directed transfer from a shared driver, so its lag should be read as "these shapes line up at this offset", not as evidence of a directed link.
- **PCMI**: Permutation conditional mutual information (PCMI) (Li et al., 2011) computes $I(Y_t; X_{t-k} \mid Y_{t-1..t-l})$ on ordinal patterns rather than the values themselves. Each series is symbolized by the Bandt-Pompe encoding (Bandt & Pompe, 2002) at embedding dimension $m = 3$ and delay $\tau = 1$ using windows ending at t so that a symbol depends only on the series up to t. The conditional mutual information is formed from the entropies: $I(Y; X \mid Z) = H(Y,Z) + H(X,Z) - H(X,Y,Z) - H(Z)$, each with the Miller-Madow bias correction $(\hat{K} - 1)/2N$ (Paninski, 2003). It estimates the CMI of the symbol sequences, with upper bound $\log m! = 1.79$ nats at $m = 3$. Its lag resolution is also limited by m, where ordinal windows at delays within m of another share source samples, so a delay of $k = 3$ leaves PCMI non-zero at lags 1 and 2.

C Attribution of Transfer Entropy Across Delays

Equation 2 assigns to each source delay the increase in transfer entropy obtained by admitting it after all shorter delays. This appendix locates that definition among its alternatives. We show that the two attributions in common use are not decompositions of $\mathrm{TE}^{X\to Y}(K,l)$, that the increment and the Shapley value both are, and that these two differ in their treatment of redundant delays in a manner that determines which is appropriate for delay estimation.

Let $N := \{1,\dots,K\}$ index the source delays, $\mathbf{X}_S := (X_{t-j})_{j\in S}$ for $S \subseteq N$, and define the set function

$$v(S) \;:=\; I\big(Y_t; \mathbf{X}_S \mid \mathbf{Y}_t^{(l)}\big), \qquad v(\emptyset) = 0, \qquad v(N) = \mathrm{TE}^{X\to Y}(K,l). \tag{11}$$

The chain rule gives $v(S \cup \{j\}) - v(S) = I(Y_t; X_{t-j} \mid \mathbf{X}_S, \mathbf{Y}_t^{(l)}) \geq 0$, so v is monotone. An attribution assigns a share φ_k of $v(N)$ to each delay, and is a *decomposition* when $\sum_{k\in N}\varphi_k = v(N)$ with $\varphi_k \geq 0$.

For a permutation π of N, let P_k^π denote the delays preceding k in π and define the marginal vector

$$m_k^\pi \;:=\; v\big(P_k^\pi \cup \{k\}\big) - v\big(P_k^\pi\big), \qquad \text{so that} \qquad \sum_{k\in N} m_k^\pi \;=\; v(N) \tag{12}$$

by telescoping, for every π. Monotonicity makes each term non-negative, so every marginal vector is a decomposition. This is the property the two common attributions lack. Scoring delays in isolation and conditioning on all remaining delays,

$$m_k(l) \;=\; v\big(\{k\}\big) - v(\emptyset), \qquad c_k(l) \;=\; v(N) - v\big(N \setminus \{k\}\big), \tag{13}$$

Table 5: Do errors in adjacent cells of the TE surface cancel when differenced? Stationary benchmark at $K = 6$, $L = 3$, same (system, seed) pairs for both estimators, exact ground truth. Across seeds, per cell: the sd of the TE estimate, the sd of the delay profile $\delta_k = \mathrm{TE}(k, l) - \mathrm{TE}(k-1, l)$, the ratio of the variance of the δ_k error to the sum of the two cell variances (1 for independent errors, below 1 when errors are positively correlated), and the pooled correlation of the two cells' errors. The null direction has true TE $= 0$ everywhere, so its errors are pure estimation noise.

direction	estimator	systems	seeds	cells	sd TE(k)	sd δ_k	Var(δ err) / (Var$_k$ + Var$_{k-1}$)	corr(err$_k$, err$_{k-1}$)
causal	ATEMPO (one fit)	5	5	75	0.011	0.008	0.286	0.715
	TENDE (one fit per cell)	5	5	75	0.019	0.026	0.947	0.053
null	ATEMPO (one fit)	5	5	75	0.001	0.000	0.222	0.782
	TENDE (one fit per cell)	5	5	75	0.001	0.001	0.715	0.300

are the marginal contributions of k when k is placed first and last respectively. Neither corresponds to a single permutation, since every delay cannot be ordered first, nor every delay last, and Eq. 12 therefore does not apply to either.

The consequence is visible whenever delays carry overlapping information, which an autocorrelated source produces at every pair of adjacent delays. Let $X_{t-1} = X_{t-2} = Z$ with $I(Y_t; Z \mid \mathbf{Y}_t^{(l)}) = 1$ nat, so $v(\{1\}) = v(\{2\}) = v(\{1, 2\}) = 1$. Then $(m_1, m_2) = (1, 1)$, which sums to twice $v(N)$ and credits the shared nat to both delays; in a recording this appears as a peak broadened across all delays adjacent to the true one. Conversely $(c_1, c_2) = (0, 0)$, which sums to zero, because each copy of Z renders the other redundant and no delay adds anything to the remainder of the history.

Among decompositions the choice is between a single ordering and an average over orderings. The increment of Eq. 2 is the marginal vector at the recency ordering $\pi_0 = (1, 2, \ldots, K)$, while the Shapley value is the mean over all orderings,

$$\delta_k(l) \;=\; m_k^{\pi_0}, \qquad \phi_k(l) \;=\; \frac{1}{K!}\sum_{\pi} m_k^{\pi} \;=\; \sum_{S \subseteq N \setminus \{k\}} \frac{|S|!\,(K - |S| - 1)!}{K!}\Big[v\big(S \cup \{k\}\big) - v(S)\Big]. \tag{14}$$

On the system above these give $(\delta_1, \delta_2) = (1, 0)$ and $(\phi_1, \phi_2) = (0.5, 0.5)$. Both are decompositions, and they differ only in how the redundant nat is divided. Averaging over orderings makes ϕ invariant to the labelling of delays, so a delay and a later repetition of it receive equal shares. Delays are not exchangeable: influence propagates forward, and information present at delay 1 and repeated at delay 2 originates at delay 1. Fixing π_0 imposes precisely this precedence, which is what permits $\boldsymbol{\delta}(l)$ to be read as a statement about when the source acts. Evaluating Eq. 14 would in any case require 2^{K-1} coalitions per delay, and at the depths used here ($K = 336$ on WEATHER-5K) no sampling scheme attains the precision a delay estimate requires.

A final consequence concerns estimation rather than definition. Since $\delta_k(l)$ differences two cells of the same surface, an estimator fitted separately per cell contributes two independent errors to every entry of the profile, whereas a single amortized network evaluates both cells with shared parameters and the errors are correlated. Table 5 measures this on the stationary benchmark, where both estimators were run on the same systems and seeds against exact ground truth. Across five seeds, the errors of adjacent cells correlate at 0.05 for per-cell fits and 0.72 for ATEMPO; the variance of the error in δ_k is 0.95 of the sum of the two cell variances for per-cell fits, as independence predicts, and 0.29 for ATEMPO. As a result the profile is noisier than the surface for per-cell fits (sd 0.026 against 0.019 nats) and less noisy than the surface for ATEMPO (0.008 against 0.011), a three-fold reduction in the sd of δ_k. Amortization therefore affects the accuracy of delay detection and not only its cost.

D SYNTHETIC DATASET RESULTS

Table 6: Relative error (%) of the TE estimate at the true lag, mean ± sd over 5 seeds. Bold is the lowest error per system.

method system	family	NPEET	NJEE	MINE	CCMI	PCMI	TENDE	ATEMPO
flat_linear	AGM-TE	6.7 ± 2.3	18.5 ± 4.5	8.5 ± 1.4	5.1 ± 5.2	67.6 ± 2.6	**3.9** ± 4.7	6.8 ± 4.1
flat_joint	AGM-TE	**2.1** ± 1.7	19.0 ± 3.7	12.1 ± 7.1	2.8 ± 2.5	76.0 ± 1.4	8.7 ± 3.0	9.7 ± 5.2
lag_linear_m3_d4	multi-dimensional	93.1 ± 5.3	38.6 ± 4.7	8.8 ± 4.2	79.4 ± 9.4	27.0 ± 6.3	6.1 ± 2.8	**3.7** ± 2.6
lag_multisource	multi-dimensional	44.9 ± 2.7	30.3 ± 5.6	7.4 ± 4.4	20.6 ± 4.5	27.2 ± 4.3	**2.5** ± 1.6	4.9 ± 0.9
lag_nonlin_m3_d4	multi-dimensional	81.8 ± 2.1	35.2 ± 3.3	18.8 ± 3.7	**8.7** ± 4.0	47.6 ± 2.2	29.6 ± 30.9	9.6 ± 3.0
lag_sq_m20	mean-blind	90.8 ± 0.4	31.8 ± 2.2	26.5 ± 2.7	20.9 ± 4.1	81.6 ± 0.8	28.7 ± 3.0	**13.4** ± 3.0
lag_vol_m20	mean-blind	72.3 ± 3.4	57.1 ± 7.8	**1.2** ± 1.4	30.6 ± 5.8	59.1 ± 3.3	34.6 ± 9.0	9.9 ± 3.3

Table 7: Signed relative error (%) at the true lag (negative = under-estimate), mean ± sd over seeds.

method system	family	true TE	NPEET	NJEE	MINE	CCMI	PCMI	TENDE	ATEMPO
flat_linear	AGM-TE	0.128	-0.6 ± 7.8	-18.5 ± 4.5	2.1 ± 9.3	-1.9 ± 7.5	-67.6 ± 2.6	-3.3 ± 5.3	2.3 ± 8.3
flat_joint	AGM-TE	0.415	-0.7 ± 2.8	-19.0 ± 3.7	-12.1 ± 7.1	-2.0 ± 3.3	-76.0 ± 1.4	-8.7 ± 3.0	-9.1 ± 6.3
lag_linear_m3_d4	multi-dim.	0.128	-93.1 ± 5.3	-38.6 ± 4.7	8.8 ± 4.2	79.4 ± 9.4	-27.0 ± 6.3	-6.1 ± 2.8	1.5 ± 4.6
lag_multisource	multi-dim.	0.128	-44.9 ± 2.7	-30.3 ± 5.6	-6.3 ± 6.2	20.6 ± 4.5	-27.2 ± 4.3	0.5 ± 3.2	2.9 ± 4.6
lag_nonlin_m3_d4	multi-dim.	0.415	-81.8 ± 2.1	-35.2 ± 3.3	-18.8 ± 3.7	8.7 ± 4.0	-47.6 ± 2.2	-29.6 ± 30.9	-9.6 ± 3.0
lag_sq_m20	mean-blind	0.988	-90.8 ± 0.4	-31.8 ± 2.2	-26.5 ± 2.7	-20.9 ± 4.1	-81.6 ± 0.8	-28.7 ± 3.0	-13.4 ± 3.0
lag_vol_m20	mean-blind	0.170	-72.3 ± 3.4	-57.1 ± 7.8	-0.4 ± 1.9	30.6 ± 5.8	-59.1 ± 3.3	-34.6 ± 9.0	-9.9 ± 3.3

Table 8: Top-1 lag recovery (successes / trials) on every synthetic system. Stationary systems: one trial per seed; time-varying systems: one trial per 1000-sample window.

family	system	trials	NPEET	NJEE	MINE	CCMI	PCMI	Liang -Kleeman	DTW	TENDE	ATEMPO
AGM-TE	flat_linear	seeds	5/5	5/5	5/5	5/5	5/5	5/5	5/5	5/5	5/5
multi-dimensional	flat_joint	seeds	5/5	5/5	5/5	5/5	5/5	2/5	5/5	5/5	5/5
	lag_linear_m3d4	seeds	1/5	5/5	5/5	5/5	5/5	5/5	5/5	5/5	5/5
	lag_multisource	seeds	5/5	5/5	5/5	5/5	5/5	3/5	5/5	5/5	5/5
	lag_nonlin_m3d4	seeds	5/5	5/5	5/5	5/5	5/5	3/5	5/5	5/5	5/5
mean-blind	lag_sq_m20	seeds	5/5	5/5	5/5	5/5	5/5	0/5	0/5	5/5	5/5
	lag_vol_m20	seeds	5/5	5/5	5/5	5/5	5/5	0/5	0/5	5/5	5/5
time-varying	tv_switch	windows	8/8	7/8	8/8	7/8	7/8	3/8	5/8	8/8	8/8
mean-blind time-varying	tv_drift	windows	8/8	5/8	7/8	7/8	6/8	5/8	7/8	7/8	6/8
	tv_null	windows	8/8	6/8	8/8	6/8	8/8	4/8	4/8	8/8	8/8
	tv_switch_sq	windows	8/8	8/8	8/8	8/8	8/8	1/8	8/8	8/8	8/8
	tv_switch_sq_wk	windows	6/8	6/8	6/8	7/8	3/8	2/8	4/8	5/8	8/8

TE Estimation Table 6 shows the TE relative error across all seven datasets with known TE, which are averaged to produce the results in Table 2. Table 7 also includes the sign of the TE estimation error, showing how some estimators like NPEET and NJEE consistently underestimate TE, whereas CCMI tends to overestimate, especially on multi-dimensional systems.

Lag Detection Accuracy Table 8 shows the Top-1-accuracy lag detection results broken down per dataset, where a success is only if the true lag has the highest ΔTE estimate. Non-time-varying lags were evaluated across five seeds, and time-varying lags were evaluated with one seed and eight windows, so a score of 6/8 means the maximum TE corresponded to the true lag in six of the eight windows.

D.1 ABLATION

Table 9 shows the full results across the three synthetic time-varying systems which were averaged to produce Table 3 and Figure 3. ATEMPO has perfect detection across all datasets and all windows. ATEMPO (no time-conditioning) is robust to window size, but performance is significantly degraded without time-conditioning. ATEMPO (per-window) shows how having to train data per-window degrades performance significantly as window size decreases. This trend is also observed across all other baselines as they too have to be trained per-window.

Table 9: Top-1 detection rate on a moving lag against samples per window, per system and averaged. Baselines are re-fitted in every window; ATEMPO is fitted once.

	switch				switch_sq				switch_sq_wk			
win_len	125	250	500	1000	125	250	500	1000	125	250	500	1000
Liang-Kleeman	0.39	0.38	0.44	0.38	0.17	0.19	0.12	0.12	0.14	0.03	0.12	0.25
NPEET	0.67	0.84	0.94	1.00	0.95	1.00	1.00	1.00	0.42	0.69	0.81	0.75
NJEE	0.19	0.25	0.69	0.88	0.69	0.88	1.00	1.00	0.17	0.19	0.19	0.75
MINE	0.22	0.41	0.69	1.00	0.61	0.94	0.94	1.00	0.12	0.22	0.38	0.75
CCMI	0.03	0.03	0.06	0.88	0.44	0.88	1.00	1.00	0.09	0.34	0.56	0.88
PCMI	0.33	0.44	0.69	0.88	0.39	0.81	1.00	1.00	0.16	0.19	0.31	0.38
DTW	0.48	0.53	0.75	0.62	0.62	0.72	1.00	1.00	0.19	0.19	0.44	0.50
TENDE	0.06	0.06	0.25	1.00	0.14	0.16	0.75	1.00	0.12	0.12	0.06	0.62
ATEMPO (no time-conditioning, per-window)	0.09	0.03	0.62	1.00	0.09	0.25	0.94	1.00	0.00	0.09	0.31	1.00
ATEMPO (no time-conditioning)	0.59	0.56	0.50	0.50	0.55	0.50	0.50	0.50	0.78	0.75	0.81	0.75
ATEMPO	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00

Table 10: Ablation arms: forecasting MSE against window length, per system.

	switch				switch_sq				switch_sq_wk			
win_len	125	250	500	1000	125	250	500	1000	125	250	500	1000
NPEET	0.224	0.228	0.208	0.209	0.313	**0.099**	**0.099**	**0.099**	0.117	0.111	0.108	0.097
NJEE	0.257	0.256	0.249	0.210	0.578	0.292	**0.099**	**0.099**	0.124	0.122	0.122	0.117
MINE	0.251	0.244	0.248	0.209	0.899	0.132	0.114	**0.099**	0.123	0.126	0.111	0.098
CCMI	0.258	0.259	0.258	**0.208**	1.142	0.372	**0.099**	**0.099**	0.122	0.121	0.117	0.093
PCMI	0.257	0.260	0.279	**0.208**	1.226	0.402	**0.099**	**0.099**	0.122	0.124	0.122	0.116
Liang-Kleeman	0.246	0.244	0.236	0.257	1.446	1.442	1.433	1.446	0.122	0.122	0.122	0.119
DTW	0.258	0.245	0.234	0.262	0.607	0.748	**0.099**	**0.099**	0.122	0.122	0.114	0.106
TENDE	0.258	0.259	0.258	0.209	1.451	1.500	0.372	**0.099**	0.122	0.122	0.122	0.114
ATEMPO (no time-conditioning, per-window)	0.255	0.257	0.236	0.209	1.451	1.449	0.336	**0.099**	0.122	0.123	0.114	**0.093**
ATEMPO (no time-conditioning)	0.245	0.263	0.272	0.247	0.954	0.955	0.955	0.955	0.103	0.107	0.100	0.115
ATEMPO	**0.209**	**0.209**	**0.209**	0.209	**0.099**	**0.099**	**0.099**	**0.099**	**0.093**	**0.093**	**0.093**	**0.093**

Table 10 extends this to compare forecasting MSE across these datasets, with the full ATEMPO selected-lags always producing the lowest MSE which is robust to window size, whereas the ablations trained per-window are only able to match this performance at the largest window size. ATEMPO (no time-conditioning) represents ATEMPO with lag amortization but time-conditioning removed and always performs worst, showing the benefit time conditioning has to downstream forecasting.

E MMTT-BENCH FULL RESULTS

Table 11: MMTT-Bench lag recovery per planted delay (3 seeds), null rejection on shuffled text, and cost relative to ATEMPO.

method	m=1	m=2	m=3	m=4	m=5	m=6	mean	shuffled (null)	time rel. ATEMPO
Liang-Kleeman	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.940	0.000
DTW	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.890	0.000
NPEET	1.000	0.000	0.000	1.000	0.000	0.000	0.330	0.890	0.050
NJEE	1.000	0.330	1.000	1.000	0.330	0.670	0.720	0.940	0.550
MINE	1.000	0.330	1.000	1.000	1.000	0.670	0.830	0.940	1.330
CCMI	1.000	0.000	0.670	1.000	1.000	1.000	0.780	0.830	0.410
PCMI	1.000	1.000	1.000	1.000	1.000	1.000	1.000	0.890	0.000
TENDE	1.000	1.000	1.000	1.000	0.670	1.000	0.940	1.000	16.280
ATEMPO	1.000	0.670	1.000	1.000	1.000	1.000	0.940	1.000	1.000

Table 12: MMTT-Bench by text condition, averaged over planted delays $m \in \{1-6\}$ and 3 seeds. Left: what ATEMPO estimates. Right: % MSE reduction over forecasting the signal from its own past alone, when one text lag is added: the benchmark's default lag 1, the lag ATEMPO selects, and a random lag. Only correct text carries a lag to find; on the three controls no choice of lag helps.

	ATEMPO estimates			% MSE		
condition	TE (nats)	peak δTE	lag found	default (lag 1)	ATEMPO lag	random lag
correct	0.52	0.27	1.00	15.59	41.26	10.37
incorrect	0.21	0.05	0.00	7.89	1.09	3.12
irrelevant	0.14	0.05	0.00	-0.69	-2.35	-1.32
shuffled	0.15	0.05	0.00	-1.50	-1.18	-1.19

Table 11 shows the full lag detection results for each planted lag and the shuffled condition where no lags are present. All these detection results are averaged to produce the column in result Table 2. ATEMPO and TENDE each only fail to detect the one shifted lag, narrowly behind PCMI which detects them all perfectly.

Table 12 shows the peak δTE values for each of the different text categories in MMTT-Bench. ATEMPO correctly reconstructs the datasets information signal, where correct contains the highest amount of information, incorrect contains less but still non-zero due to the symmetry of the time series, and irrelevant and shuffles represent the null where the text contributes no additional information.

F WEATHER5K RESULTS

Table 13: WEATHER-5K, results using ridge forecaster model on the 300-station subset, reported as MSE per variable for every lag selector, candidate pool and budget. Random rows are means over $n = 20$ draws with the sd of the normalized metric. Δ is against the 48 h default (240 features).

inputs	n feat.	temp.	dew point	wind dir.	wind speed	pres-sure	MSE norm.	sd norm.	Δ (%)
matched (67)									
Random	40	20.9	21.6	9086	5.04	53.8	0.507	0.008	1.17
	60	19.4	20.9	8956	4.87	52.4	0.494	0.006	-1.51
	80	18.7	20.5	8869	4.79	51.4	0.486	0.006	-3.07
	120	18.3	20.3	8800	4.71	51.3	0.481	0.004	-3.97
ATEMPO	40	20.4	22.0	9056	5.08	52.9	0.506	–	0.839
	60	19.7	21.4	8990	5.00	52.3	0.499	–	-0.483
	80	19.2	21.0	8939	4.93	51.9	0.494	–	-1.51
	120	18.3	20.2	8813	4.75	50.7	0.481	–	-3.99
TENDE	40	25.7	26.1	9247	5.20	64.3	0.552	–	10.1
	60	24.8	25.9	9165	5.11	64.2	0.546	–	8.88
	80	24.1	25.7	9117	5.05	64.0	0.542	–	8.02
	120	23.3	25.3	9039	4.98	63.5	0.535	–	6.78
MINE	40	23.3	22.8	9161	5.16	57.2	0.525	–	4.63
	60	20.5	22.2	9066	4.98	56.5	0.512	–	2.08
	80	20.1	21.9	9009	4.91	55.5	0.505	–	0.762
	120	19.5	21.7	8929	4.83	55.3	0.500	–	-0.235
CCMI	40	20.0	21.6	9022	5.02	52.3	0.501	–	-0.0762
	60	19.6	21.4	8972	4.96	52.0	0.497	–	-0.932
	80	18.9	20.9	8906	4.85	51.4	0.489	–	-2.40
	120	18.1	20.1	8761	4.69	50.5	0.478	–	-4.72
NJEE	40	20.1	21.1	9011	4.99	52.2	0.499	–	-0.474
	60	19.2	20.8	8923	4.88	51.6	0.491	–	-2.03
	80	18.6	20.5	8852	4.79	51.1	0.485	–	-3.30
	120	18.0	20.0	8747	4.67	50.5	0.477	–	-4.94
NPEET	40	22.2	21.2	9021	5.07	51.9	0.503	–	0.368
	60	19.5	21.0	8935	4.91	51.7	0.493	–	-1.68
	80	18.8	20.4	8864	4.80	50.8	0.485	–	-3.26
	120	18.3	20.3	8793	4.74	50.8	0.481	–	-4.10
PCMI	40	33.2	23.9	9276	5.59	59.4	0.558	–	11.2
	60	32.9	23.4	9222	5.52	58.8	0.552	–	10.1
	80	22.2	20.7	8971	5.03	51.9	0.500	–	-0.202
	120	18.7	20.4	8841	4.77	51.3	0.484	–	-3.36
Liang-Kleeman	40	26.6	25.1	9235	5.55	54.9	0.539	–	7.42
	60	23.0	23.9	9138	5.31	54.1	0.522	–	4.12
	80	21.7	23.7	9097	5.23	53.9	0.517	–	3.14
	120	21.2	23.7	9062	5.21	54.0	0.515	–	2.72
DTW	40	22.7	23.8	9131	5.30	54.1	0.521	–	3.98
	60	21.3	23.7	9083	5.21	54.0	0.516	–	2.86
	80	20.4	22.6	9016	5.08	53.0	0.506	–	0.898
	120	19.6	21.8	8949	4.98	52.4	0.498	–	-0.628
full (336)									

Continued on next page

Table 13: WEATHER-5K, ridge forecaster on the 300-station subset, test MSE per variable for every lag selector, candidate pool and budget (same ridge, λ rule and test rows throughout). Random rows are means over draws with the sd of the normalised metric. Δ is against the 48 h default (240 features).

inputs	n feat.	temp.	dew point	wind dir.	wind speed	pres-sure	MSE norm.	sd norm.	Δ (%)
Default (48 h)	240	19.9	22.4	8939	5.01	53.0	0.501	–	0.00
Random	40	23.0	23.5	9181	5.12	58.7	0.528	0.018	5.39
	60	20.9	22.3	9063	4.96	56.1	0.511	0.017	1.95
	80	20.3	22.1	8995	4.89	55.9	0.506	0.016	0.943
	120	19.5	21.5	8895	4.79	54.7	0.497	0.016	-0.947
	240	18.3	20.6	8730	4.64	52.5	0.481	0.012	-4.01
most recent n	40	26.6	25.1	9235	5.55	54.9	0.539	–	7.42
	60	23.3	24.5	9169	5.40	54.6	0.528	–	5.23
	80	22.0	24.2	9124	5.30	54.3	0.521	–	3.96
	120	21.2	23.7	9062	5.21	54.0	0.515	–	2.72
	240	19.9	22.4	8939	5.01	53.0	0.501	–	0.00
ATEMPO	40	22.9	22.1	9108	5.19	53.2	0.514	–	2.45
	60	19.5	21.1	8986	4.96	51.8	0.496	–	-1.08
	80	19.1	20.9	8935	4.90	51.8	0.492	–	-1.81
	120	18.3	20.3	8825	4.76	51.0	0.482	–	-3.76
	240	17.7	19.9	8684	4.62	50.4	0.473	–	-5.63
MINE	40	22.3	23.2	9186	5.12	58.6	0.527	–	5.11
	60	21.2	22.8	9083	4.98	58.2	0.518	–	3.31
	80	21.0	22.7	9042	4.94	58.1	0.515	–	2.81
	120	20.4	22.6	8964	4.86	58.0	0.510	–	1.82
	240	19.2	21.7	8809	4.72	55.8	0.496	–	-1.15
CCMI	40	20.9	22.7	9069	5.14	53.1	0.510	–	1.63
	60	19.3	21.3	8968	4.95	51.8	0.495	–	-1.20
	80	18.6	20.5	8883	4.82	51.0	0.486	–	-3.03
	120	18.1	20.2	8790	4.71	50.7	0.479	–	-4.36
	240	17.6	19.9	8646	4.58	50.3	0.471	–	-6.10
NJEE	40	23.0	23.9	9224	5.17	59.0	0.531	–	6.01
	60	22.2	23.7	9163	5.08	58.8	0.526	–	4.98
	80	19.5	20.7	8919	4.81	51.6	0.489	–	-2.43
	120	18.8	20.4	8842	4.73	51.2	0.483	–	-3.56
	240	18.0	20.0	8700	4.61	50.6	0.474	–	-5.38
NPEET	40	22.9	23.3	9126	5.27	53.6	0.519	–	3.47
	60	20.8	23.1	9052	5.14	53.4	0.510	–	1.81
	80	20.1	22.2	8999	5.03	52.7	0.503	–	0.238
	120	19.7	22.1	8950	4.98	52.6	0.499	–	-0.427
	240	18.5	20.8	8808	4.78	51.4	0.485	–	-3.30
PCMI	40	33.2	23.9	9276	5.59	59.4	0.558	–	11.2
	60	32.9	23.4	9222	5.52	58.8	0.552	–	10.1
	80	22.2	20.7	8971	5.03	51.9	0.500	–	-0.202
	120	19.0	20.3	8814	4.77	51.3	0.484	–	-3.39
	240	18.0	20.1	8683	4.61	51.1	0.475	–	-5.18
Liang-Kleeman	40	26.6	25.1	9235	5.55	54.9	0.539	–	7.42
	60	23.0	23.9	9138	5.31	54.1	0.522	–	4.12
	80	21.7	23.7	9097	5.23	53.9	0.517	–	3.14

Continued on next page

Table 13: WEATHER-5K, ridge forecaster on the 300-station subset, test MSE per variable for every lag selector, candidate pool and budget (same ridge, λ rule and test rows throughout). Random rows are means over draws with the sd of the normalised metric. Δ is against the 48 h default (240 features).

inputs	n feat.	temp.	dew point	wind dir.	wind speed	pres-sure	MSE norm.	sd norm.	Δ (%)
	120	21.2	23.7	9062	5.21	54.0	0.515	–	2.72
	240	18.8	20.8	8849	4.83	51.4	0.487	–	-2.81
DTW	40	22.7	23.8	9131	5.30	54.1	0.521	–	3.98
	60	21.3	23.7	9083	5.21	54.0	0.516	–	2.86
	80	20.4	22.6	9013	5.08	53.0	0.506	–	0.876
	120	19.9	22.5	8958	5.01	53.0	0.502	–	0.0922
	240	18.5	20.8	8795	4.78	51.5	0.485	–	-3.32

We follow the WEATHER-5K benchmark and report MSE in the original units across five features. We use two different forecasting model architectures, iTransformer as our main benchmark comparison in Table 14, and a ridge regression in Table 13.

Using the simpler linear ridge model on WEATHER5K 300-station subset enables a larger number of experiments which would not have been possible using the more computationally expensive iTransformer architecture. Table 13 shows that despite its simplicity, the ridge regression model performs well. ATEMPO's improvements of 3.8% between the 240 feature WEATHER5K baseline recent(48h) and the 120 feature ATEMPO features remains comparable to the 4.6% improvement using iTransformer, showing the improvement from ATEMPO selected lags is robust across models.

Further baselines were run, however TENDE was computationally intractable to run even across a 300 station subset, so instead we feed them a small sub-selected number of candidate lags, $n = 67$, designed in a grid which is dense at short lags and logarithmically sparser at long ones. Because a 67-candidate search is strictly easier than a 336-candidate one, we ran all baselines across both for a fair comparison.

Table 14: WEATHER-5K, iTransformer, test MSE per variable in original units (pred_len 168). 'MSE norm.' is the benchmark's scaled-space metric; Δ is its % change against the 48 h default of the same station set.

inputs	n feat.	seed	steps (x1000)	selection (h)	train (min)	temp.	dew point	wind dir.	wind speed	pressure	MSE norm.	Δ (%)
(a) all 5,672 stations												
ATEMPO	240	2021	300	1.787	79	18.373	21.828	8830.489	5.180	59.022	0.516	-7.254
Default (48 h)	240	2021	300	0.000	110	20.406	24.516	9374.630	5.613	63.687	0.556	0.000
(b) 300 stations												
ATEMPO	240	2021	200	1.787	79	16.959	19.999	8672.705	4.559	58.924	0.491	-5.951
		2022	200	1.787	49	17.063	20.065	8687.295	4.563	58.946	0.491	-5.821
		2023	200	1.787	48	17.061	20.113	8682.039	4.558	59.163	0.492	-5.759
		2024	200	1.787	49	17.055	20.050	8675.365	4.563	58.590	0.490	-6.034
		2025	200	1.787	49	16.833	19.868	8670.379	4.552	58.052	0.488	-6.454
CCMI	240	2021	200	4.358	49	16.814	19.806	8664.002	4.543	59.308	0.491	-5.976
		2022	200	4.358	49	16.771	19.753	8651.829	4.516	59.140	0.489	-6.272
		2023	200	4.358	48	16.867	19.948	8654.402	4.542	59.362	0.491	-5.946
		2024	200	4.358	49	16.833	19.764	8661.952	4.557	59.512	0.491	-5.823
		2025	200	4.358	48	16.795	19.828	8655.864	4.540	59.342	0.490	-6.007
DTW	240	2021	200	0.186	59	17.875	21.107	8827.293	4.696	59.604	0.502	-3.804
		2022	200	0.186	60	17.814	21.027	8820.951	4.698	60.140	0.503	-3.609
		2023	200	0.186	61	17.751	20.996	8825.594	4.703	59.553	0.502	-3.853
Liang	240	2021	200	0.495	59	18.084	21.069	8892.234	4.736	58.896	0.503	-3.623
		2022	200	0.495	63	18.075	21.110	8906.986	4.730	58.955	0.503	-3.562
		2023	200	0.495	62	18.040	21.019	8896.114	4.738	60.129	0.506	-3.061
MINE	240	2021	200	4.632	49	18.444	21.784	8876.295	4.722	66.527	0.522	-0.017
		2022	200	4.632	51	18.497	21.973	8877.440	4.722	66.122	0.521	-0.141
		2023	200	4.632	50	18.521	21.897	8895.313	4.725	67.203	0.524	0.422
NJEE	240	2021	200	2.942	48	17.378	20.193	8705.982	4.586	60.687	0.497	-4.729
		2022	200	2.942	50	17.304	20.090	8706.281	4.576	60.268	0.496	-5.014
		2023	200	2.942	50	17.303	20.190	8709.997	4.579	61.502	0.499	-4.395
NPEET	240	2021	200	2.596	59	17.817	20.972	8849.000	4.693	59.042	0.501	-4.043
		2022	200	2.596	60	17.797	20.930	8848.920	4.693	59.201	0.501	-3.987
		2023	200	2.596	61	17.751	20.910	8843.000	4.702	58.815	0.500	-4.153
PCMI	240	2021	200	1.289	48	17.212	20.229	8717.397	4.640	60.391	0.498	-4.555
		2022	200	1.289	50	17.258	20.356	8726.672	4.625	60.597	0.498	-4.459
		2023	200	1.289	50	17.344	20.377	8732.143	4.657	61.495	0.502	-3.835
Random	240	2021	200	0.000	73	17.680	20.812	8802.316	4.647	61.911	0.505	-3.238
		2022	200	0.000	48	17.608	20.819	8805.193	4.656	62.792	0.507	-2.794
		2023	200	0.000	49	17.719	20.941	8798.976	4.655	62.997	0.508	-2.671
Default (48 h)	240	2021	200	0.000	110	18.962	22.596	9175.466	4.903	61.157	0.522	0.000
		2022	200	0.000	94	18.937	22.573	9180.363	4.904	61.258	0.522	0.055
		2023	200	0.000	108	19.060	22.643	9171.314	4.911	61.582	0.523	0.252
ATEMPO	120	2021	100	1.787	48	17.189	20.334	8925.412	4.697	56.133	0.494	-5.358
		2022	200	1.787	48	17.442	20.549	8930.164	4.697	58.555	0.500	-4.123
		2023	200	1.787	57	17.441	20.437	8914.127	4.698	59.371	0.502	-3.836
		2024	200	1.787	49	17.384	20.415	8918.404	4.701	56.767	0.496	-5.012
		2025	200	1.787	48	17.291	20.446	8922.896	4.707	58.760	0.501	-4.062
CCMI	120	2021	200	4.358	48	17.297	20.298	8846.284	4.649	56.961	0.493	-5.531
		2022	200	4.358	49	17.377	20.355	8852.358	4.660	57.790	0.495	-5.039
		2023	200	4.358	48	17.358	20.394	8846.122	4.659	57.696	0.495	-5.102
		2024	200	4.358	49	17.270	20.330	8847.415	4.660	57.626	0.495	-5.165
		2025	200	4.358	48	17.383	20.458	8852.969	4.658	57.779	0.496	-5.025
MINE	120	2021	200	4.632	49	19.693	23.010	9078.167	4.870	69.391	0.540	3.466
		2022	200	4.632	49	19.664	22.955	9069.425	4.867	69.517	0.540	3.453
		2023	200	4.632	48	19.692	22.889	9071.364	4.859	69.343	0.539	3.326
NJEE	120	2021	200	2.942	47	18.055	20.730	8913.436	4.716	61.334	0.508	-2.637
		2022	200	2.942	48	18.029	20.828	8915.019	4.703	61.620	0.508	-2.552
		2023	200	2.942	47	17.845	20.625	8911.631	4.715	62.101	0.509	-2.370
NPEET	120	2021	200	2.596	77	19.058	22.648	9279.161	4.963	61.940	0.528	1.117
		2022	200	2.596	78	19.074	22.642	9286.872	4.968	61.033	0.526	0.765
		2023	200	2.596	89	19.067	22.656	9286.169	4.964	61.500	0.527	0.950
PCMI	120	2021	200	1.289	48	18.199	20.716	8905.024	4.780	59.776	0.506	-2.999
		2022	200	1.289	49	18.222	20.763	8906.501	4.773	59.921	0.506	-2.951
		2023	200	1.289	48	18.204	20.735	8902.049	4.785	60.007	0.507	-2.874
Random	120	2021	100	0.000	47	18.357	21.364	8992.654	4.765	62.660	0.515	-1.227
		2022	200	0.000	49	18.596	21.561	9003.798	4.776	63.630	0.519	-0.578
		2023	200	0.000	47	18.511	21.550	8995.024	4.802	63.530	0.519	-0.537
Default (24 h)	120	2021	200	0.000	166	21.082	24.930	10021.410	5.373	65.506	0.568	8.878
		2022	200	0.000	112	21.108	24.974	10031.114	5.377	65.483	0.568	8.943
		2023	200	0.000	110	21.073	24.932	10020.902	5.367	65.454	0.568	8.818

Table 15: Where each selector spends its feature budget: the lag sets the iTransformer arms were trained on, counted by band. Default refers to the most recent n =budget lags.

budget	selector	persis -tence	sub- daily	diurnal	synoptic	weekly	boundary	phase R	median lag (h)
24	Default	3	19	2	0	0	0	0.00	12.50
	Random	0	2	0	21	1	0	0.29	162.00
	PCMI	1	3	13	6	0	1	0.35	114.00
	NPEET	2	13	1	8	0	0	0.11	17.50
	MINE	0	0	2	20	2	0	0.20	154.50
	NJEE	1	0	3	18	0	2	0.11	254.50
	ATEMPO	2	7	3	5	1	6	0.26	54.00
	CCMI	2	9	2	11	0	0	0.18	40.00
48	Default	3	19	5	21	0	0	0.00	24.50
	Random	0	4	4	35	3	2	0.19	147.50
	Liang-Kleeman	3	19	12	14	0	0	0.32	24.50
	DTW	3	19	7	19	0	0	0.11	24.50
	PCMI	1	5	13	27	1	1	0.23	136.50
	NPEET	3	16	5	24	0	0	0.10	28.50
	MINE	0	1	5	40	2	0	0.07	170.00
	NJEE	2	0	4	36	1	5	0.11	241.00
	CCMI	2	11	7	28	0	0	0.12	111.00
	ATEMPO	3	7	11	19	1	7	0.30	118.00

Table 16: Which of ATEMPO's selected lags do the forecasting work. Ridge on 300 stations, scored using mean MSE over variables. Each band row removes that band from ATEMPO's set and reports the % change of the metric relative to the full set: a positive *cost of band* means the forecast got worse without it, i.e. the band was helping; *per lag (band)* divides by the band's size, and the leave-one-out (LOO) columns give the mean and the largest change from removing a single lag of the band.

budget	band	n lags	cost of band (%)	per lag (band)	per lag (LOO mean)	per lag (LOO max)
24	persistence	2	1.54	0.77	0.30	0.55
	sub-daily	7	1.42	0.20	0.05	0.11
	diurnal	3	0.43	0.14	0.11	0.12
	synoptic	4	1.18	0.30	0.20	0.33
	weekly	1	0.86	0.86	0.86	0.86
	boundary	7	0.28	0.04	0.01	0.04
48	persistence	3	1.44	0.48	0.18	0.52
	sub-daily	7	0.66	0.09	0.02	0.06
	diurnal	11	0.59	0.05	0.03	0.06
	synoptic	18	2.19	0.12	0.04	0.14
	weekly	1	0.14	0.14	0.14	0.14
	boundary	8	0.09	0.01	0.00	0.02

F.1 Physical analysis of selected lags

WEATHER5K is not labelled with ground truth lags, however weather data does have natural patterns we can use to compare the selected lags across baselines. Specifically, we classify each selector's lags into persistence (the most recent 1-3 hours), sub-daily (within 24 hours), diurnal harmonics (multiples of 24hours, ± 1), synoptic (everything else between 1-14 days), weekly, and boundary band (the final seven hours 330-336). Phase R is the circular concentration of the lags on the 24 h clock (1 = every lag at the same time of day), and random is the single draw the random arm used. We used the 24-lag sets that were used to train an iTransformer model, with full forecasting results available in Appendix F.

Table 15 shows the results. NPEET only selects lags occuring in the first 59 hours. PCMI over-selects lags that are all spaced 24 hours apart, which can make physical sense if a certain hour of

day is the strongest driving signal, but also likely increases the amount of redundant information contained in the lag sub-set. MINE and NJEE forecasting performance are indistinguishable from random and this is supported as most of their lags are not well distributed and have no obvious pattern apart from being in the first 1-14 days. ATEMPO is the only estimator that spreads lags across all our categories, reflecting it's strong ability to select lags that maximize information for downstream forecasting.

However ATEMPO does select a surprisingly high number of lags in the final 6 hours, where all other estimators select zero. We perform an ablation that drops each lag band in turn from ATEMPO's chosen lags to see the effect on forecasting performance using the ridge model. Table 16 shows the change in MSE when removing a lag from a specific band. *cost of band* is the change in Δ from removing it, so positive means the set got worse and the band was helping. The leave-one-out (LOO) columns give the mean and the largest cost of removing a single lag of the band. The boundary lags are consistently the least valuable, with the smallest positive cost across all three columns. Removing any band removes information, so we would always expect a small positive change.

Our hypothesis is the high number of boundary layer lags may be an artifact of how the delay profile is calculated when $k \to K$. It can be understood through the geometry of the lag encoding; TE is estimated from Eq. 9, whose inner term is the squared score difference between the two conditionings. These differ in exactly k coordinates and therefore have a Euclidean distance of $\sqrt{k}$, resulting in approximation errors that de-correlate as k increases. This can produce a residual error that also grows with k. This artefact would then create the largest effect at large k, corresponding the boundary errors observed as $k \to K$.